Data Assimilation –
Theoretical and Algorithmic Aspects

Lars Nerger

Alfred Wegener Institute for Polar and Marine Research
Bremerhaven, Germany

and
Bremen Supercomputing Competence Center BremHLR
Bremen, Germany

Lars.Nerger@awi.de

BremHLR
Kompetenzzentrum für Höchstleistungsrechnen Bremen

AWI

KIAPS, May 28, 2013

Overview

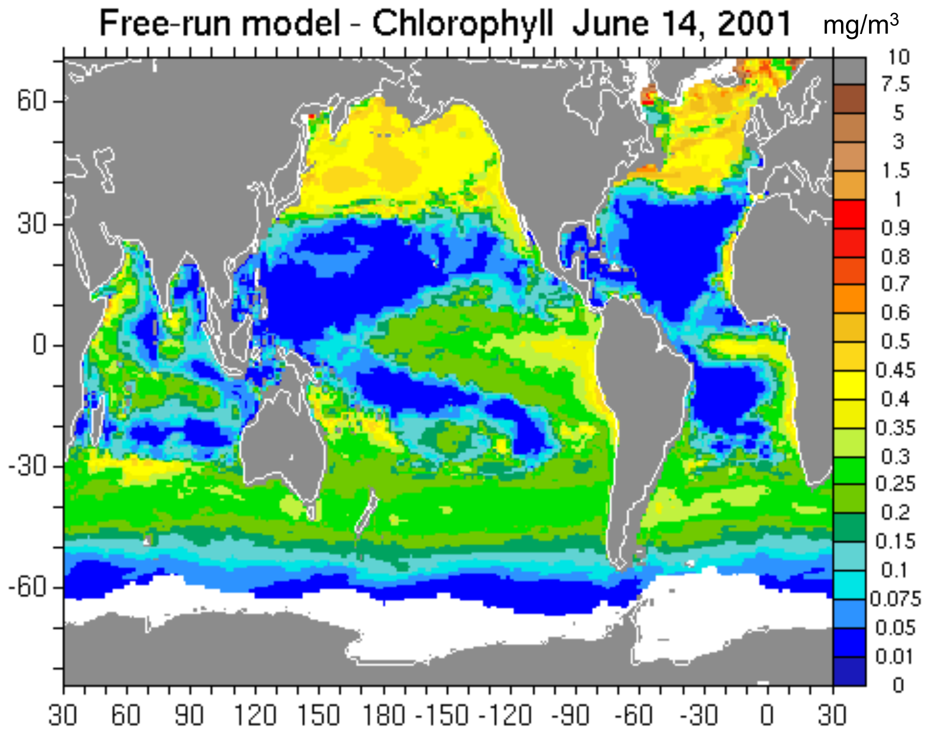
Data Assimilation algorithms –
where are we and how did we get here?

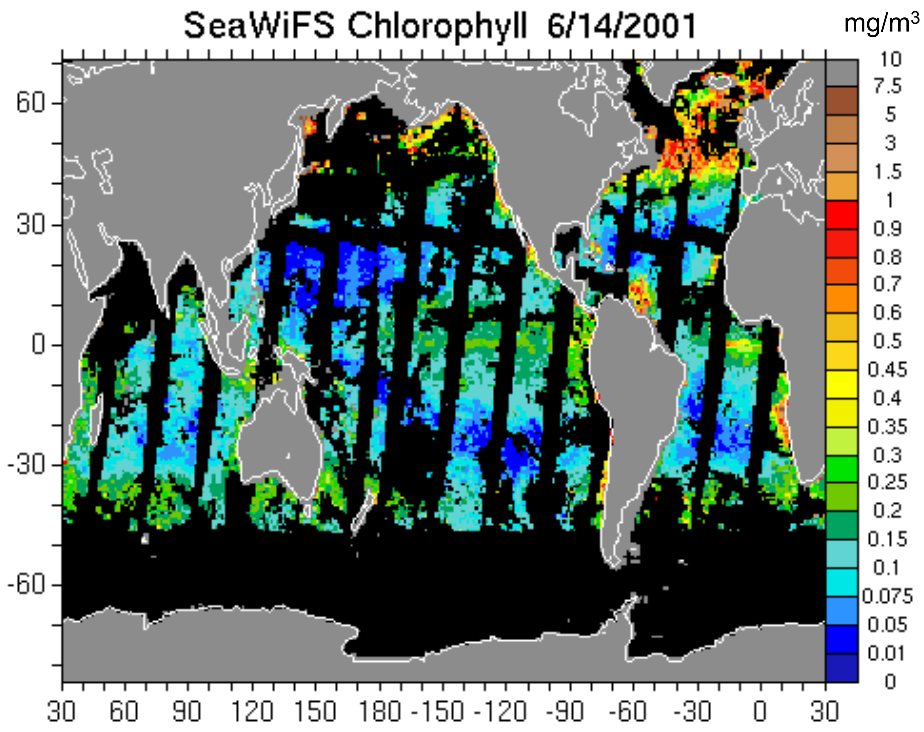
A review – with focus on ensemble data assimilation

- Data assimilation problem
- Variational data assimilation
- Sequential data assimilation
- Ensemble Kalman Filters
- Ensemble Square-root Filters
- Nonlinearity & current developments

Data Assimilation

Example: Chlorophyll in the ocean





Information: Model

- Generally correct, but has errors
- all fields, fluxes, …

Information: Observation

- Generally correct, but has errors
- sparse information
  (only surface, data gaps, one field)

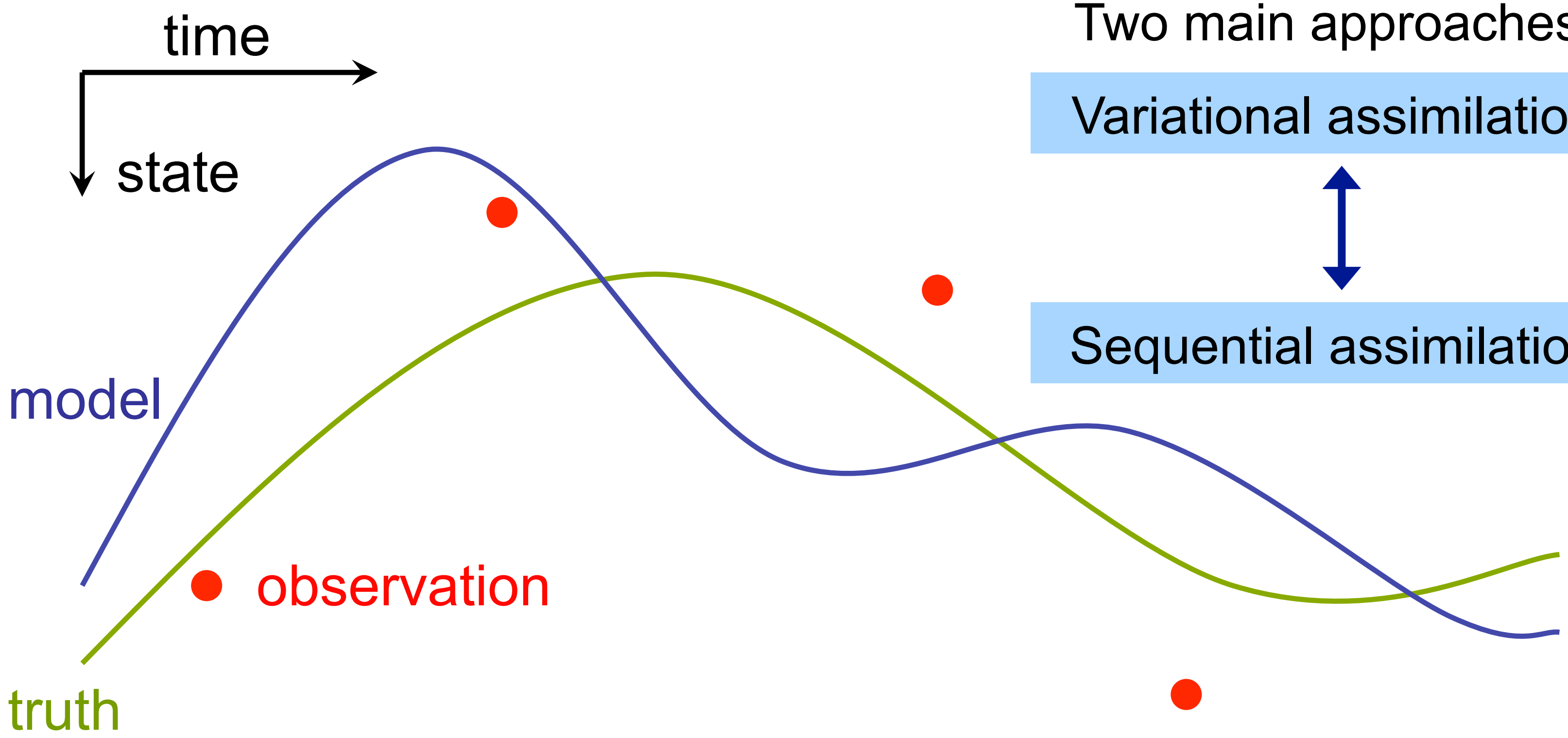
Data Assimilation

- Optimal estimation of system state:
  - initial conditions (for weather/ocean forecasts, …)
  - state trajectory (temperature, concentrations, …)
  - parameters (growth of phytoplankton, …)
  - fluxes (heat, primary production, …)
  - boundary conditions and ‘forcing’ (wind stress, …)

- Characteristics of system:
  - high-dimensional numerical model - $\mathcal{O}(10^7\text{-}10^9)$
  - sparse observations
  - non-linear

Data Assimilation

Consider some physical system (ocean, atmosphere,…)



Two main approaches:

Variational assimilation

↕

Sequential assimilation

Optimal estimate basically by least-squares fitting

Data Assimilation – Model and Observations

Two components:

1. State: $\mathbf{x} \in \mathbb{R}^n$

   Dynamical model

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

2. Obervations: $\mathbf{y} \in \mathbb{R}^m$

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

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

Some views on Data Assimilation

Data Assimilation – an inverse problem

Model provides a background state $\mathbf{x}^b$ (prior knowledge)

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

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

at some time instance

Now solve for state:

$$\mathbf{x} = \mathbf{x}^b + H^{-1}\left[y - H\left[\mathbf{x}^b\right]\right]$$

Issues:

- Compute H^{-1} - or pseudo inverse $\left(H^T H\right)^{-1} H^T$
- Inversion could be possible with regularization
- This formulation ignores model and observation errors

Data Assimilation – least squares fitting

Background state $\mathbf{x}^b \in \mathbb{R}^n$

Weight matrices (acknowledge different uncertainties):

$\mathbf{B}$ for background state

$\mathbf{R}$ for observations

"Cost function":

$$J(\mathbf{x}) = (\mathbf{x}-\mathbf{x}^b)^T\mathbf{B}^{-1}(\mathbf{x}-\mathbf{x}^b) + (\mathbf{y}-H[\mathbf{x}])^T\mathbf{R}^{-1}(\mathbf{y}-H[\mathbf{x}])$$

Background   Observations

Optimal $\tilde{\mathbf{x}}$ minimizes J:

Requiring $\mathrm{dJ/dx} = 0$ leads to:

$$\tilde{\mathbf{x}} = \mathbf{x}^b + \mathbf{B}H^T\left(H\mathbf{B}H^T + \mathbf{R}\right)^{-1}\left(\mathbf{y} - H\mathbf{x}^b\right)$$

No explicit statistical assumptions required!

Optimal Interpolation (OI)

1. Parameterize (prescribe) matrices $\mathbf{B}$ and $\mathbf{R}$
   (e.g. by using estimated decorrelation lengths)
2. Compute the optimal (variance-minimizing) state $\tilde{\mathbf{x}}$ as

$$\tilde{\mathbf{x}} = \mathbf{x}^b + \mathbf{B}H^T \left(H\mathbf{B}H^T + \mathbf{R}\right)^{-1} \left(\mathbf{y} - H\mathbf{x}^b\right)$$

OI was quite common about 20-30 years ago.

Several issues:

- Parameterized matrices
- Computing cost
- Optimality of solution

Data Assimilation – an estimation problem

Probability density of $\mathbf{x}$: $p\left(\mathbf{x}_i\right)$

Probability density of $\mathbf{y}$: $p\left(\mathbf{y}_i\right)$

Likelihood of $\mathbf{y}$ given $\mathbf{x}$: $p\left(\mathbf{y}_i | \mathbf{x}_i\right)$

Bayes law: Probability density of $\mathbf{x}$ given $\mathbf{y}$

$$p\left(\mathbf{x}_i | \mathbf{y}_i\right) = \frac{p\left(\mathbf{y}_i | \mathbf{x}_i\right) p\left(\mathbf{x}_i\right)}{p\left(\mathbf{y}_i\right)}$$

With *prior knowledge*:

Probability of $\mathbf{x}_i$ given all observations $\mathbf{Y}_i$ up to time i

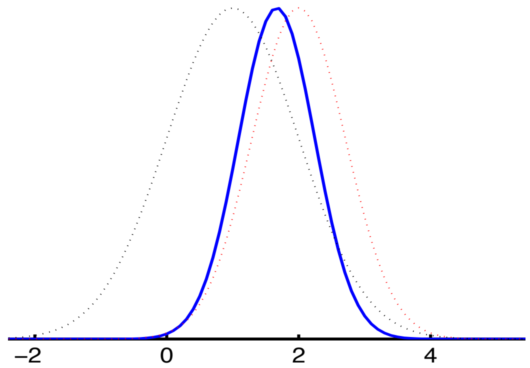
$$p\left(\mathbf{x}_i | \mathbf{Y}_i\right) = \frac{p\left(\mathbf{y}_i | \mathbf{x}_i\right) p\left(\mathbf{x}_i | \mathbf{Y}_{i-1}\right)}{p\left(\mathbf{y}_i | \mathbf{Y}_{i-1}\right)}$$

Data Assimilation – Probabilistic Assumptions

Assume Gaussian distributions:

$$\mathcal{N}\left(\mu, \sigma^2\right) = a\ e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

(fully described by mean and variance)



Observations: $\mathcal{N}(\mathbf{y}, \mathbf{R})$

State: $\mathcal{N}(\mathbf{x}, \mathbf{P})$

Posterior state distribution

$$p(\mathbf{x}_i | \mathbf{Y}_i) \sim a e^{-J(\mathbf{x})}$$

With

$$J(\mathbf{x}) = (\mathbf{x} - \mathbf{x}^b)^T \mathbf{P}^{-1} (\mathbf{x} - \mathbf{x}^b) + (\mathbf{y} - H[\mathbf{x}])^T \mathbf{R}^{-1} (\mathbf{y} - H[\mathbf{x}])$$

(same as for least squares – there are statistical assumptions!)

Variational Data Assimilation

3D-Var, 4D-Var, Adjoint Method

Variational Data Assimilation

- Based on optimal control theory
- Examples: "adjoint method", "4D-Var", "3D-Var"

- Method:
  - 1. Formulate "cost function"

$$J(\mathbf{x}_0) = \sum_{i=1}^{k} \underbrace{\left(\mathbf{x}_i - \mathbf{x}_i^b\right)^T \mathbf{C} \left(\mathbf{x}_i - \mathbf{x}_i^b\right)}_{\text{Background}} + \underbrace{\left(\mathbf{y}_i - H\mathbf{x}_i\right)^T \mathbf{D} \left(\mathbf{y}_i - H\mathbf{x}_i\right)}_{\text{Observation}}$$

  - 2. Minimize cost (by variational method)

⇨ 3D-Var: Do this locally in time for each i

x: model state
x^b: background
y: observation
i: time index
C, D: weight matrices

Adjoint Method - 4D-Var

- formulate cost J in terms of "control variable"
  Example: initial state x_0
- Problem:
  Find trajectory (defined by x_0) that minimizes cost J while fulfilling model dynamics
- Use gradient-based algorithm:
  - e.g. quasi-Newton
  - Gradient for $J[x_0]$ is computed using adjoint of tangent linear model operator
  - The art is to formulate the adjoint model (No closed formulation of model operator)
  - Iterative procedure (local in control space)

Adjoint method - 4D-Var algorithm

1. Initialization: Choose initial estimate of x_0

2. Forward: Integrate model
start from x_0; store trajectory

3. Compute cost function
exit, if cost is below limit

4. Backward: Integrate adjoint model backward in time
start from final residual (y-x); use trajectory from 2.

5. Optimizer: Update x_0
with optimization algorithm

Issues of 4D-Var/3D-Var

- Coding of adjoint model
- Computing cost
  - Method is iterative, limited parallelization possibilities
- Storage requirements
  - Store full forward trajectory
- Limited size of time window in case of nonlinear model
- Parameterized weight matrices

Sequential Data Assimilation

Kalman filters

Error propagation

Linear stochastic dynamical model

$$\mathbf{x}_i = \mathbf{M}_{i-1,i}\mathbf{x}_{i-1} + \boldsymbol{\eta}_i$$

Assume that $p(\mathbf{x}_{i-1}) = \mathcal{N}\left(\mathbf{x}_{i-1}, \mathbf{P}^a_{i-1}\right)$

Also assume uncorrelated state errors and model errors $\boldsymbol{\eta}_i$

Then

$$\mathbf{P}^f_i = \mathbf{M}_{i-1,i}\mathbf{P}^a_{i-1}(\mathbf{M}_{i-1,i})^T + \mathbf{Q}_{i-1}$$

With model error covariance matrix $\mathbf{Q}_{i-1}$

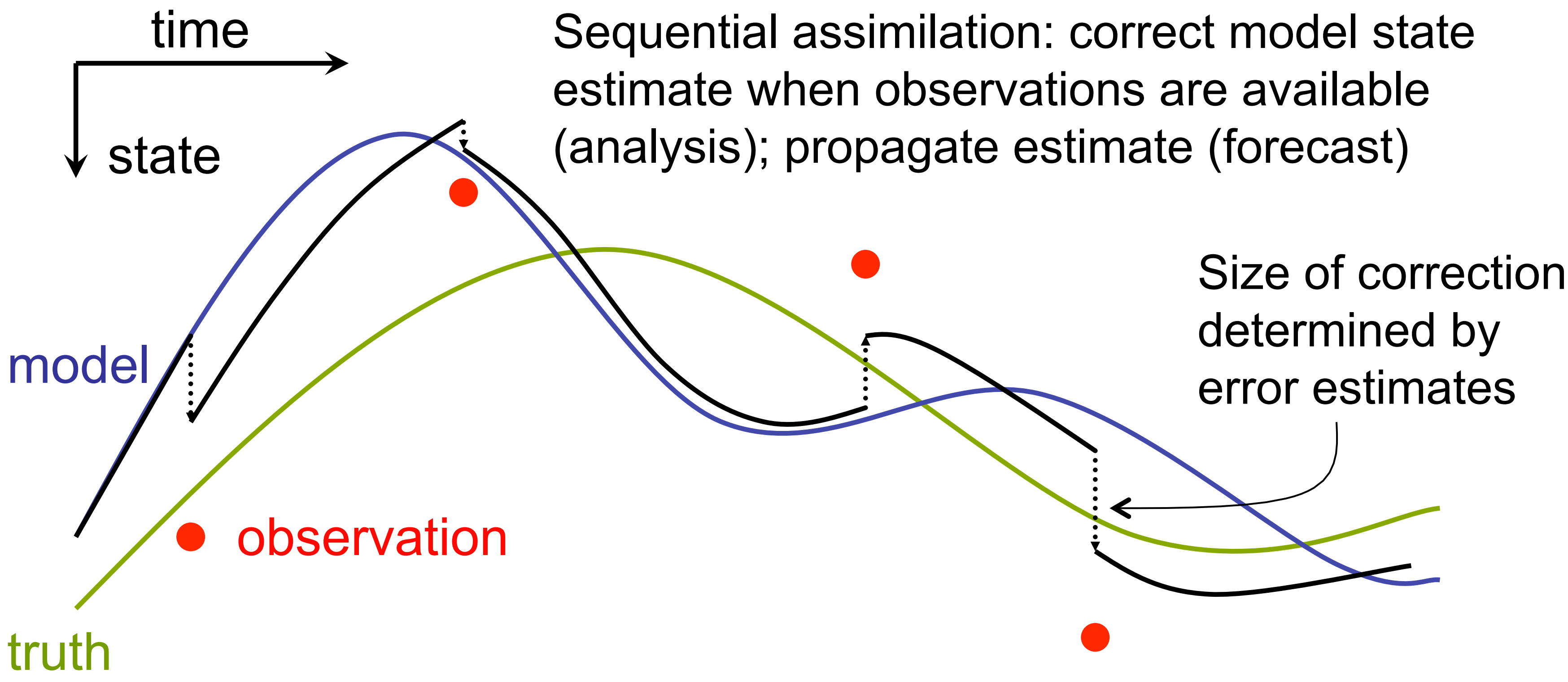
Error propagation builds the foundation of the Kalman filter

More later…

Sequential Data Assimilation

Consider some physical system (ocean, atmosphere,…)

Sequential assimilation: correct model state estimate when observations are available (analysis); propagate estimate (forecast)

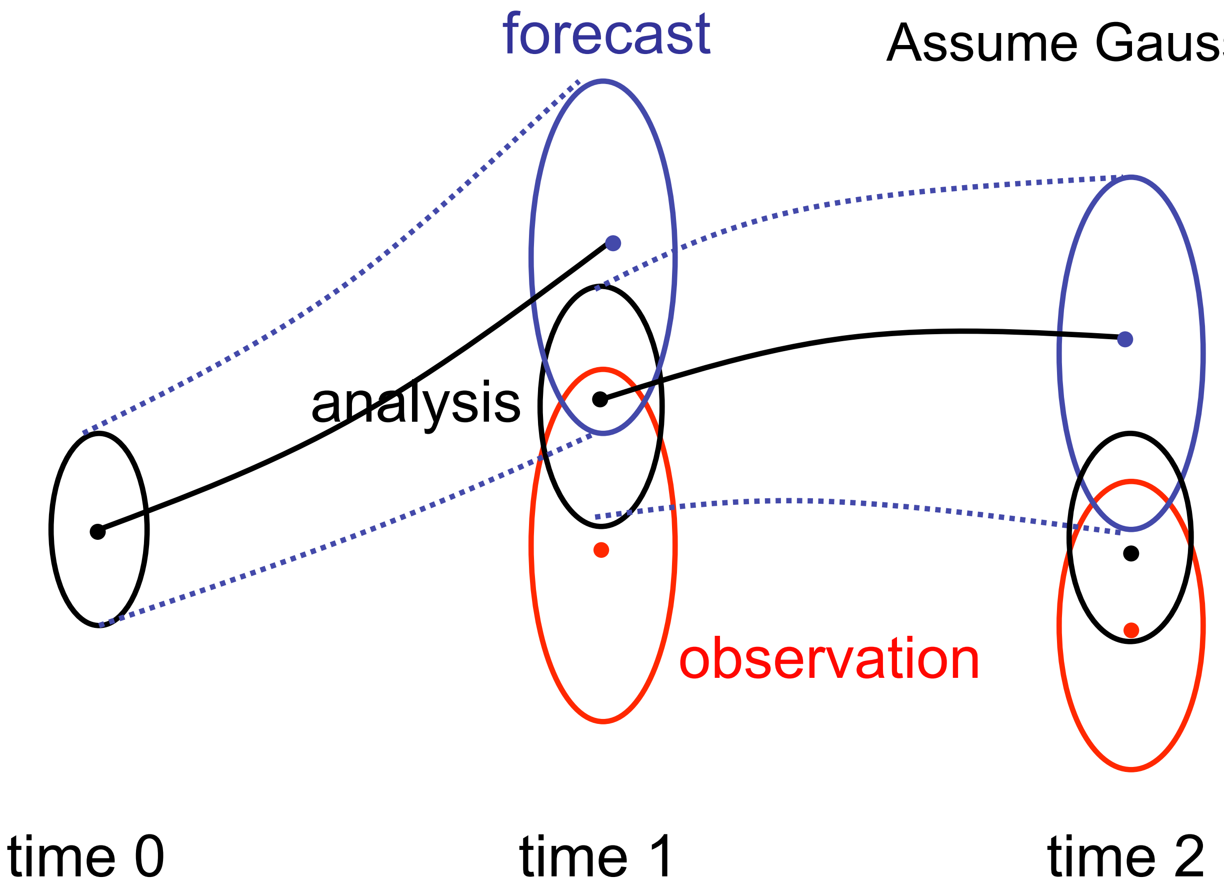


3D-Var is “sequential” but usually not called like it

Probabilistic view: Optimal estimation

Consider probability distribution of model and observations

Kalman Filter:
Assume Gaussian distributions



The Kalman Filter

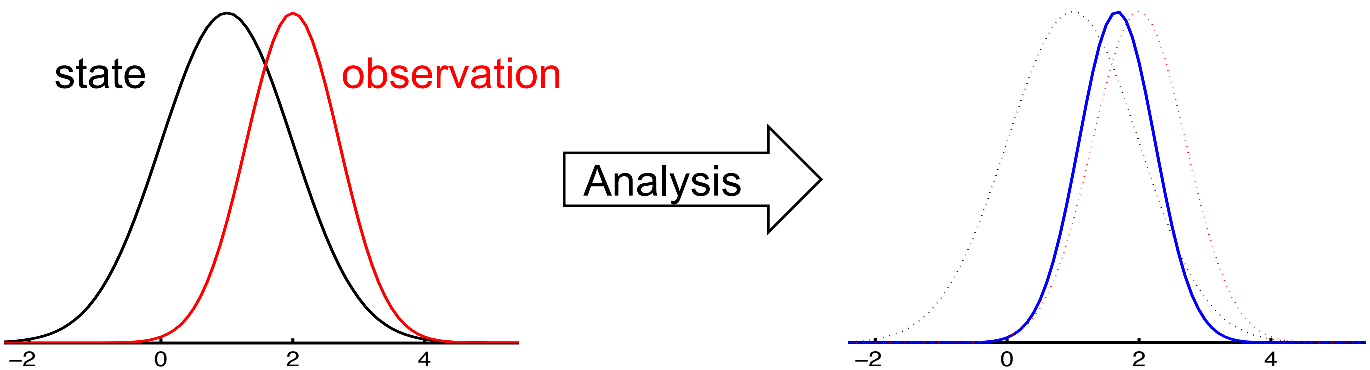
Assume Gaussian distributions
fully described by

- mean state estimate
- covariance matrix

➜ Strong simplification of estimation problem

Analysis is combination auf two Gaussian distributions
computed as

- Correction of state estimate
- Update of covariance matrix



Kalman Filter (Kalman, 1960)

Forecast:

State propagation

$$\mathbf{x}_i = \mathbf{M}_{i-1,i}\mathbf{x}_{i-1} + \epsilon_i$$

Propagation of error estimate

$$\mathbf{P}_i^f = \mathbf{M}_{i-1,i}\mathbf{P}_{i-1}^a(\mathbf{M}_{i-1,i})^T + \mathbf{Q}_{i-1}$$

Analysis at time t_k:

State update

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k^f\right)$$

Update of error estimate

$$\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{P}_k^f$$

with “Kalman gain”

$$\mathbf{K}_k = \mathbf{P}_k^f\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}$$

The KF (Kalman, 1960)

Initialization: Choose initial state estimate **x** and corresponding covariance matrix **P**

↓

Forecast: Evolve state estimate with model. Evolve columns/rows of covariance matrix with model.

↓

Analysis: Combine state estimate with observations based on weights computed from error estimates of state estimate and observations. Update matrix **P** according to relative error estimates.

The KF (Kalman, 1960)

With nonlinear model: Extended Kalman filter

Initialization: Choose initial state estimate **x** and corresponding covariance matrix **P**

↓

Forecast: Evolve state estimate with non-linear model. Evolve columns/rows of covariance matrix with linearized model.

↓

Analysis: Combine state estimate with observations based on weights computed from error estimates of state estimate and observations. Update matrix **P** according to relative error estimates.

Issues of the Kalman Filter

- Storage of covariance matrix can be unfeasible (n^2 with n of $\mathcal{O}(10^7\text{-}10^9)$)
- Evolution of covariance matrix extremely costly
- Linearized evolution (like in Extended KF) can be unstable (e.g. Evensen 1992, 1993)
- Adjoint model $\mathbf{M}_{i-1,i}^T$ can be avoided using

$$\mathbf{M}_{i-1,i}\left(\mathbf{M}_{i-1,i}\mathbf{P}_{i-1}^a\right)^T$$

⇨ Need to reduce the cost

“Suboptimal” Filters

Approaches to reduce the cost of the Kalman filter

- Simplified error evolution
  (constant, variance only)
- Reduce rank of **P**
- Reduce resolution of model
  (at least for the error propagation)
- Reduce model complexity

Examples:

- „suboptimal schemes“, Todling & Cohn 1994
- Approximate KF, Fukumori & Malanotte, 1995
- RRSQRT, Verlaan & Heemink, 1995/97
- SEEK, Pham et al., 1998

Low-rank approximation of P

Example: **SEEK filter** (Pham et al., 1998)

Approximate $\mathbf{P}_i^a \approx \mathbf{V}_i \mathbf{U}_i \mathbf{V}_i^T$

(truncated eigendecomposition)

Mode matrix $\mathbf{V}_i$ has size $n \times r$ $\mathbf{U}_i$ has size $r \times r$

Forecast of r „modes“:

$$\mathbf{V}_{i+1} = \mathbf{M}_{i,i+1} \mathbf{V}_i$$

for nonlinear model

$$\mathbf{V}_{i+1} \approx M_{i,i+1} \left(\mathbf{V}_i + [\mathbf{x}_i^a, \ldots, \mathbf{x}_i^a] \right) - M_{i,i+1} [\mathbf{x}_i^a, \ldots, \mathbf{x}_i^a]$$

Now use in analysis step:

$$\tilde{\mathbf{P}}_k^f \approx \mathbf{V}_k \mathbf{U}_{k-1} \mathbf{V}_k^T$$

The SEEK filter (Pham, 1998)

Initialization: Approximate covariance matrix by low-rank matrix in the form $\mathbf{P}=\mathbf{VUV}^T$. Choose state $\mathbf{x}$.

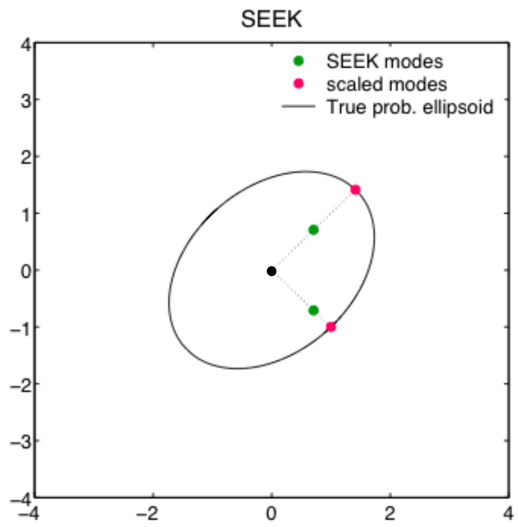
Forecast: Evolve state estimate with non-linear model. Evolve modes **V** of covariance matrix with linearized model.

Analysis: Apply EKF update step to ensemble mean and the „eigenvalue matrix“ **U**. Covariance matrix represented by modes and **U**.

Re-Initialization: Occasionally perform re-orthogonalization of modes of covariance matrix

Sampling Example

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



General sampling of probability distribution

Approximation in SEEK based on Gaussian distribution

More general:

- Sample $p(\mathbf{x})$ by N random state realizations $\mathbf{x}^{(j)}$:

$$p(\mathbf{x}) = \frac{1}{N}\sum_{j=1}^{N}\delta(\mathbf{x}-\mathbf{x}^{(j)})$$

- State **ensemble**

$$\mathbf{X} = \left[\mathbf{x}^{(1)},\ldots,\mathbf{x}^{(N)}\right]$$

- Ensemble mean $\bar{\mathbf{x}} = \frac{1}{N}\sum_{j=1}^{N}\mathbf{x}^{(j)}$

Ensemble representation (approximation) of P

Approximate

$$\mathbf{P}_i^a \approx \frac{1}{N-1} \left(\mathbf{X}_i - \bar{\mathbf{X}}_i\right) \left(\mathbf{X}_i - \bar{\mathbf{X}}_i\right)^T$$

($\bar{\mathbf{X}}_i$ holds ensemble mean in each column)

Forecast of N ensemble states:

$$\mathbf{X}_{i+1}^f = \mathbf{M}_{i,i+1}\mathbf{X}_{i+1}^a$$

for nonlinear model

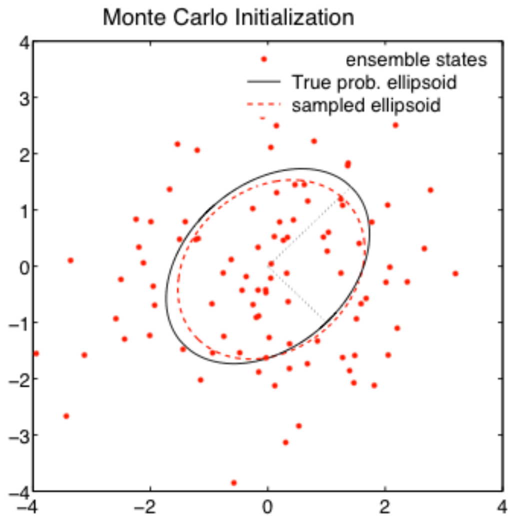
$$\mathbf{X}_{i+1}^f = M_{i,i+1}\mathbf{X}_{i+1}^a$$

Now use in analysis step:

$$\hat{\mathbf{P}}_i^f \approx \frac{1}{N-1} \left(\mathbf{X}_i^f - \bar{\mathbf{X}}_i^f\right) \left(\mathbf{X}_i^f - \bar{\mathbf{X}}_i^f\right)^T$$

Sampling Example

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



More on sampling

- Ensemble is not unique
- Gaussian assumption simplifies sampling (covariance matrix & mean state)

Example: 2nd-order exact sampling (Pham et al. 1998)

Use $$\mathbf{P}_i^a \approx \mathbf{V}_i \mathbf{S}_i \mathbf{V}_i^T$$

(truncated eigendecomposition)

Create ensemble states as

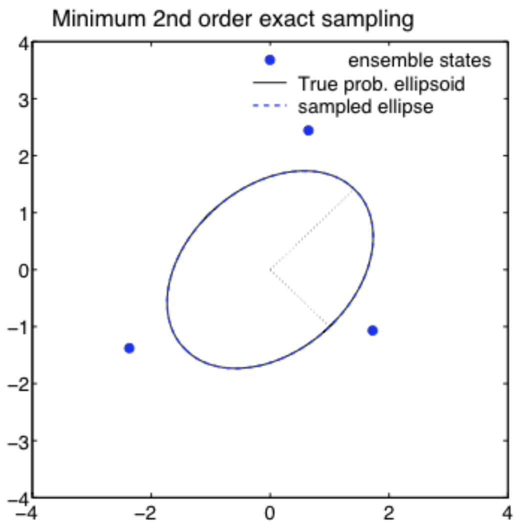
$$\mathbf{X} = \bar{\mathbf{X}} + \sqrt{N-1}\mathbf{V}\mathbf{S}^{1/2}\boldsymbol{\Omega}^T$$

$\boldsymbol{\Omega}$ is random matrix with columns orthonormal and orthogonal to vector $(1,\ldots,1)^T$. Size $N \times (N-1)$

Ensemble size $N = r + 1$

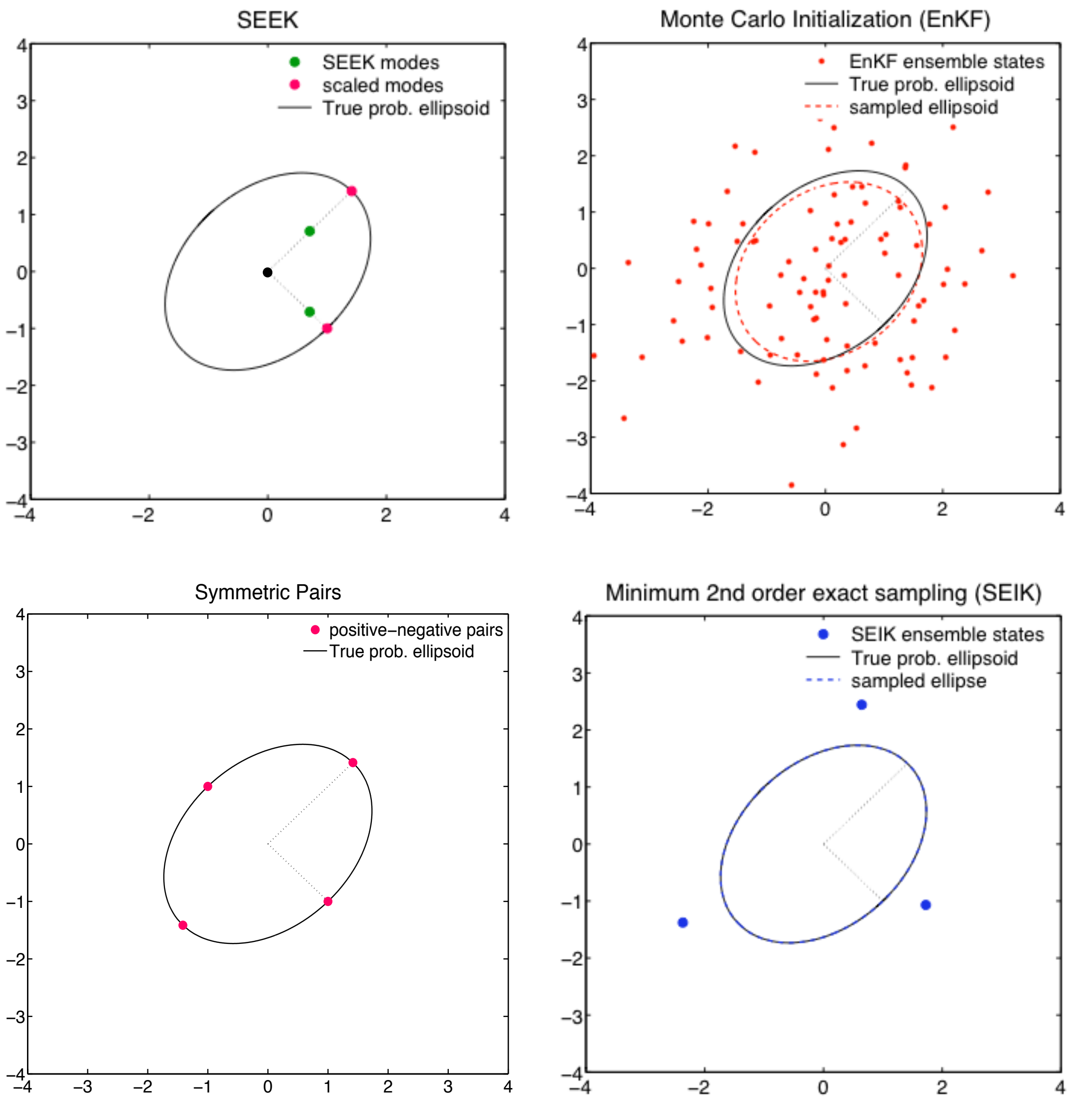
Sampling Example

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



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

Collection of possible samplings



Error Subspace Algorithms

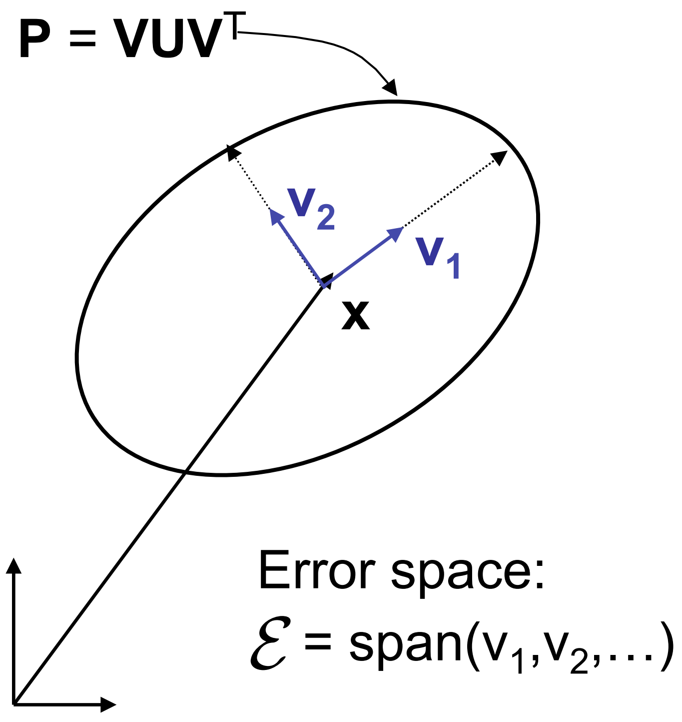
⇨ Approximate state covariance matrix by low-rank matrix

⇨ Keep matrix in decomposed form ($\mathbf{XX}^T$, $\mathbf{VUV}^T$)

Mathematical motivation:

- state error covariance matrix represents error space at location of state estimate
- directions of different uncertainty
- consider only directions with largest errors (error subspace)

⇒ degrees of freedom for state correction in analysis: rank($\mathbf{P}$)



Ensemble-based Kalman filters

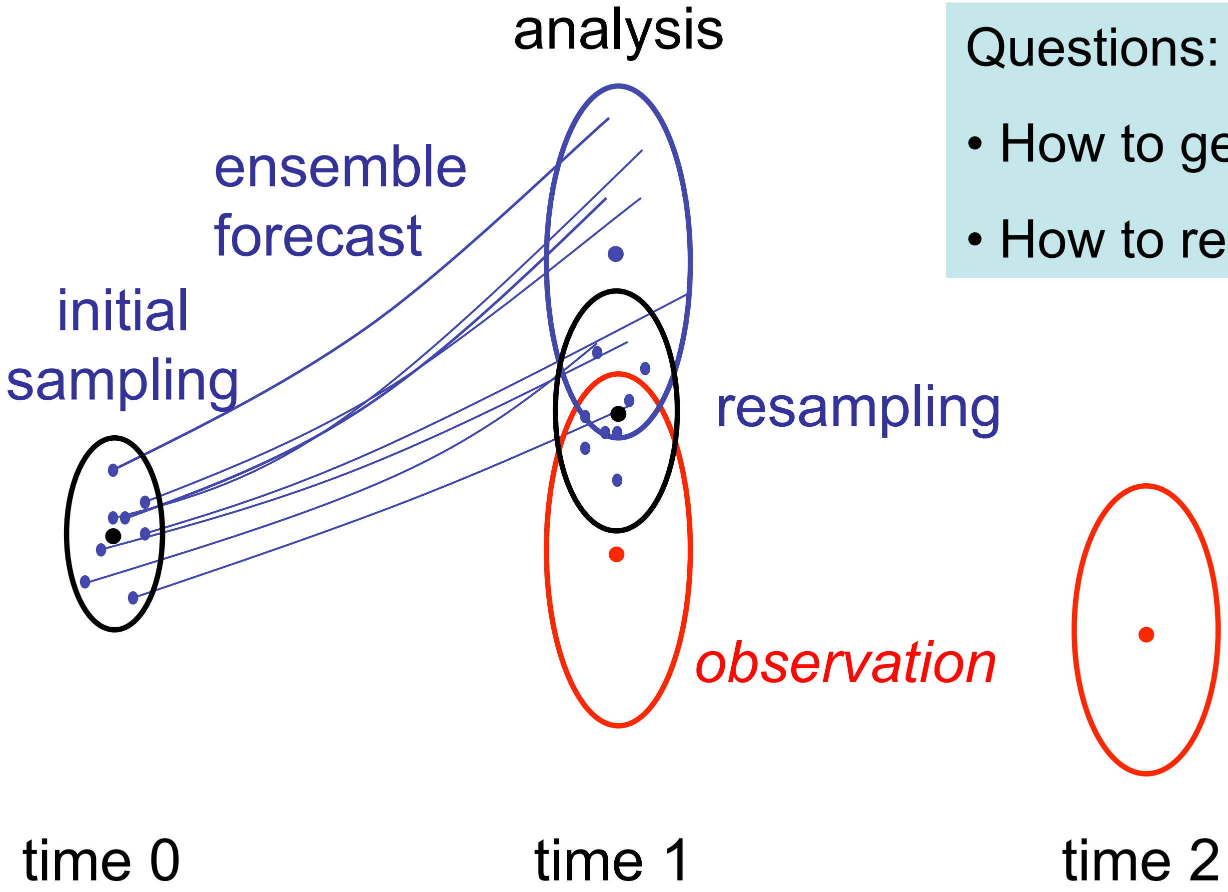
Ensemble-based Kalman Filters

- Foundation: Kalman filter (Kalman, 1960)
  - optimal estimation problem
  - express problem in terms of state estimate **x** and error covariance matrix **P** (normal distributions)
  - propagate matrix **P** by linear (linearized) model
  - variance-minimizing analysis
- Ensemble-based Kalman filter:
  - sample state **x** and covariance matrix **P** by ensemble of model states
  - propagate **x** and **P** by integration of ensemble states
  - Apply linear analysis of Kalman filter

First filter in oceanography: "Ensemble Kalman Filter" (Evensen, 1994), second: SEIK (Pham et al., 1998)

Ensemble-based Kalman Filter

Approximate probability distributions by ensembles



Questions:

- How to generate initial ensemble?
- How to resample after analysis?

Please note:

In general, this is **not an approximation** of the Kalman filter!

Efficient use of ensembles

$\mathbf{P}_k^f$ can be approximated by ensemble or modes: $\tilde{\mathbf{P}}_k^f$

Analysis at time t_k:

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \tilde{\mathbf{K}}_k \left(\mathbf{y}_k - \mathbf{H}_k \mathbf{x}_k^f \right)$$

Kalman gain

$$\tilde{\mathbf{K}}_k = \tilde{\mathbf{P}}_k^f \mathbf{H}_k^T \left(\mathbf{H}_k \tilde{\mathbf{P}}_k^f \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1}$$

Costly inversion: $m \times m$ matrix!

Ensembles allow for cost reduction – if **R** is invertible at low cost

Efficient use of ensembles (2)

Kalman gain

$$\tilde{\mathbf{K}}_k = \tilde{\mathbf{P}}_k^f \mathbf{H}_k^T \left(\mathbf{H}_k \tilde{\mathbf{P}}_k^f \mathbf{H}_k^T + \mathbf{R}_k \right)^{-1}$$

Alternative form (Sherman-Morrison-Woodbury matrix identity)

$$\tilde{\mathbf{K}}_k = \left[\left(\tilde{\mathbf{P}}_k^f \right)^{-1} + \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} \right]^{-1} \mathbf{H}^T \mathbf{R}^{-1}$$

Looks worse: $n \times n$ matrices need inversion

However: with ensemble $\tilde{\mathbf{P}}_k^f = (N-1)^{-1} \mathbf{X}^{'} \mathbf{X}^{'T}$

$$\tilde{\mathbf{K}}_k = \mathbf{X}^{'} \left[(N-1)\mathbf{I} + \mathbf{X}^{'T} \mathbf{H}^T \mathbf{R}^{-1} \mathbf{H} \mathbf{X}^{'} \right]^{-1} \mathbf{X}^{'T} \mathbf{H}^T \mathbf{R}^{-1}$$

Inversion of $N \times N$ matrix

(Ensemble perturbation matrix $\mathbf{X}^{'} = \mathbf{X} - \bar{\mathbf{X}}$)

Ensemble transformations

$\mathbf{P}_k^f$ can be approximated by ensemble or modes: $\tilde{\mathbf{P}}_k^f$

Analysis at time t_k:

State update

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \tilde{\mathbf{K}}_k \left(\mathbf{y}_k - \mathbf{H}_k \mathbf{x}_k^f\right)$$

Update of error estimate

$$\tilde{\mathbf{P}}_k^a = \left(\mathbf{I} - \tilde{\mathbf{K}}_k \mathbf{H}_k\right) \tilde{\mathbf{P}}_k^f$$

This is incomplete!

We are missing the analysis ensemble $\mathbf{X}_k^a$

Ensemble transformations (2)

Possibilities to obtain $\mathbf{X}_k^a$

1. **Monte Carlo analysis update**
    - Kalman update of each single ensemble member

2. **Explicit ensemble transformation**
    1. Kalman update of ensemble mean state
    2. Transformation of ensemble perturbations $\mathbf{X}' = \mathbf{X} - \bar{\mathbf{X}}$
        a. Right sided: $\mathbf{X}'^a = \mathbf{X}'^f \mathbf{W}$
        b. Left sided: $\mathbf{X}'^a = \hat{\mathbf{W}} \mathbf{X}'^f$

Monte Carlo analysis update

Used in Ensemble Kalman Filter (EnKF, Evensen 1994)

- Forecast ensemble $\mathbf{X}_k^f$
- Generate observation ensemble

$$\mathbf{y}^{(j)} = \mathbf{y} + \epsilon^{(j)}$$

- Update each ensemble member

$$\mathbf{X}_k^a = \mathbf{X}_k^f + \tilde{\mathbf{K}}_k \left(\mathbf{Y}_k - \mathbf{H}_k \mathbf{X}_k^f \right)$$

Pro:

- Simple implementation

Issues:

- Generation of observation ensemble
- Introduction of sampling noise through $\epsilon^{(j)}$

Right sided ensemble transformation

$$\mathbf{X}'^{a} = \mathbf{X}'^{f}\mathbf{W}$$

Used in:

- SEIK (Singular Evolutive Interpolated KF, Pham et al. 1998)
- ETKF (Ensemble Transform KF, Bishop et al. 2001)
- EnsRF (Ensemble Square-root Filter, Whitaker/Hamill 2001)

Very efficient: $\mathbf{W}$ is small ($N \times N$)

Ensemble Transform Kalman Filter - ETKF

Ensemble perturbation matrix | size

$$\mathbf{X}'_k := \mathbf{X}_k - \overline{\mathbf{X}_k}$$ (n x N)

Analysis covariance matrix

$$\mathbf{P}^a = \mathbf{X}'^f \mathbf{A} (\mathbf{X}'^f)^T$$ (n x n)

"Transform matrix" (in ensemble space)

$$\mathbf{A}^{-1} := (N-1)\mathbf{I} + (\mathbf{H}\mathbf{X}'^f)^T \mathbf{R}^{-1} \mathbf{H}\mathbf{X}'^f$$ (N x N)

Ensemble transformation

$$\mathbf{X}'^a = \mathbf{X}'^f \ \mathbf{W}^{ETKF}.$$ (n x N)

Ensemble weight matrix

$$\mathbf{W}^{ETKF} := \sqrt{N-1}\mathbf{C}\mathbf{\Lambda}$$ (N x N)

- $\mathbf{C}\mathbf{C}^T = \mathbf{A}$ (symmetric square root)
- $\mathbf{\Lambda}$ is identity or random orthogonal matrix with EV $(1, \ldots, 1)^T$)

SEIK Filter

Error-subspace basis matrix

size

$$\mathbf{L} := \mathbf{X}^f \, \mathbf{T}$$ (n x N-1)

(T subtracts ensemble mean and removes last column)

Analysis covariance matrix

$$\tilde{\mathbf{P}}^a = \mathbf{L}\tilde{\mathbf{A}}\mathbf{L}^T$$ (n x n)

“Transform matrix” (in error subspace)

$$\tilde{\mathbf{A}}^{-1} := (N-1)\mathbf{T}^T\mathbf{T} + (\mathbf{HL})^T\mathbf{R}^{-1}\mathbf{HL}$$ (N-1 x N-1)

Ensemble transformation

$$\mathbf{X}'^a = \mathbf{L}\,\mathbf{W}^{SEIK}$$ (n x N)

Ensemble weight matrix

$$\mathbf{W}^{SEIK} := \sqrt{N-1}\tilde{\mathbf{C}}\mathbf{\Omega}^T$$ (N-1 x N)

- $\tilde{\mathrm{C}}$ is square root of $\tilde{\mathrm{A}}$ (originally Cholesky decomposition)
- Ω^T is transformation from N-1 to N (random or deterministic)

The SEIK filter (Pham, 1998)

Initialization: Approximate covariance matrix by low-rank matrix in the form $\mathbf{P}=\mathbf{V}\mathbf{U}\mathbf{V}^T$. Generate ensemble of minimum size exactly representing error statistics.

Forecast: Evolve each of the ensemble members with the full non-linear stochastic model.

Analysis: Apply EKF update step to ensemble mean and the „eigenvalue matrix“ **U**. Covariance matrix approx. by ensemble statistics.

Ensemble transformation: Transform state ensemble to exactly represent updated error statistics.

Computations in ensemble-spanned space

Square root of covariance matrix (ensemble size *N,* state dim *n*)

$$\mathbf{Z} = \mathbf{X}^f \mathbf{T} \qquad\qquad \mathbf{P}^f = \mathbf{Z}\mathbf{Z}^T$$

$\mathbf{T}$ is specific for filter algorithm:

ETKF:

$\mathbf{T}$ removes ensemble mean

(usually, compute directly $\mathbf{Z} = \mathbf{X} - \overline{\mathbf{X}}$)

$\mathbf{Z}$ has dimension *nN*

SEIK:

$\mathbf{T}$ removes ensemble mean and drops last column

$\mathbf{Z}$ has dimension *n(N-1)*

Computations in ensemble-spanned space

Square root of covariance matrix (ensemble size *N*, state dim *n*)

$$\mathbf{Z} = \mathbf{X}^f\mathbf{T} \qquad \mathbf{P}^f = \mathbf{Z}\mathbf{Z}^T$$

Transformation matrix in ensemble space (small matrix)

$$\mathbf{A} = \left(\mathbf{G} + (\mathbf{HZ})^T\mathbf{R}^{-1}\mathbf{HZ}\right)^{-1}$$

ETKF:

- $\mathbf{A}$ has dimension N^2
- $\mathbf{G} = \mathbf{I}$ (identity matrix)

SEIK:

- $\mathbf{A}$ has dimension $(N-1)^2$
- $\mathbf{G} = (\mathbf{T}\,\mathbf{T}^T)^{-1}$

Computations in ensemble-spanned space

Square root of covariance matrix (ensemble size *N*, state dim *n*)

$$\mathbf{Z} = \mathbf{X}^f \mathbf{T} \qquad \mathbf{P}^f = \mathbf{Z}\mathbf{Z}^T$$

Transformation matrix in ensemble space (small matrix)

$$\mathbf{A} = \left(\mathbf{G} + (\mathbf{HZ})^T \mathbf{R}^{-1} \mathbf{HZ}\right)^{-1}$$

Analysis state covariance matrix

$$\mathbf{P}^a = \mathbf{Z}\mathbf{A}\mathbf{Z}^T$$

Computations in ensemble-spanned space

Square root of covariance matrix (ensemble size *N,* state dim *n*)

$$\mathbf{Z} = \mathbf{X}^f \mathbf{T} \qquad\qquad \mathbf{P}^f = \mathbf{Z}\mathbf{Z}^T$$

Transformation matrix in ensemble space (small matrix)

$$\mathbf{A} = \left(\mathbf{G} + (\mathbf{HZ})^T \mathbf{R}^{-1} \mathbf{HZ}\right)^{-1}$$

Analysis state covariance matrix

$$\mathbf{P}^a = \mathbf{Z}\mathbf{A}\mathbf{Z}^T$$

Ensemble transformation based on square root of $\mathbf{A}$

$$\mathbf{X}^a \sim \mathbf{Z}\mathbf{L} \qquad\qquad \mathbf{L}\mathbf{L}^T = \mathbf{A}$$

Very efficient:
Transformation matrix computed in space of dim. *N or N-1*

The SEIK filter - Properties

- Computational complexity
  - linear in dimension of state vector
  - approx. linear in dimension of observation vector
  - cubic with ensemble size
- Low complexity due to explicit consideration of error subspace:
  - ⇨ Degrees of freedom given by ensemble size -1
  - ⇨ Analysis increment: combination of ensemble members with weight computed in error subspace
- Simple application to non-linear models due to ensemble forecasts (e.g. no adjoint model)

ETKF: Practically the same properties, but analysis in ensemble space, dimension *N*

Left sided ensemble transformation

$$\mathbf{X}'^{a} = \hat{\mathbf{W}}\mathbf{X}'^{f}$$

Used in:

- EAKF (Ensemble Adjustment KF, Anderson 2001)

Issue:

- Costly in plain form: $\hat{\mathbf{W}}$ is huge ($n \times n$)
- But: Computation can be done stepwise avoiding to compute $\hat{\mathbf{W}}$

Analysis step and ensemble transformation

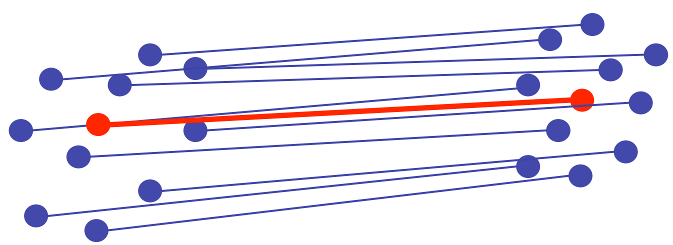
Analysis step of square-root filters:

1. correct state estimate
2. transform ensemble (forecast → analysis)

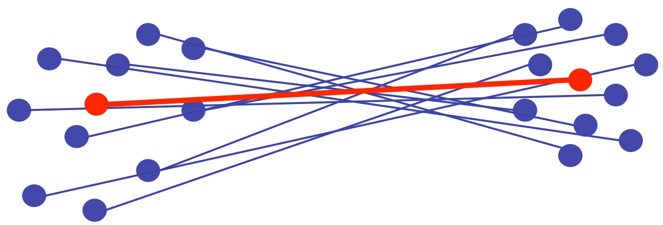
(both can be combined into a single operation)

Key element: Transformation matrix and its square-root

- Computed in space spanned by the ensemble members
- Not unique!

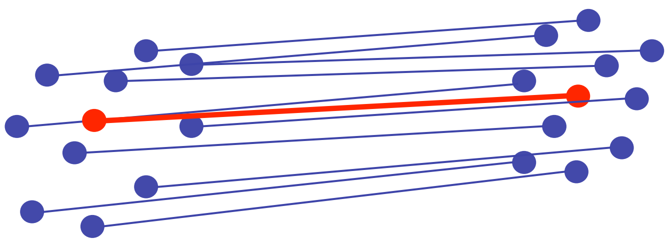


Deterministic transformation

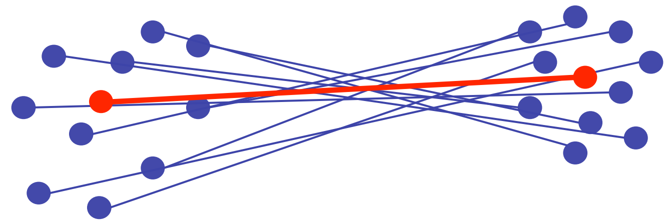


Random transformation with constraints

Ensemble transformations



Minimum transformation
(standard in ETKF)

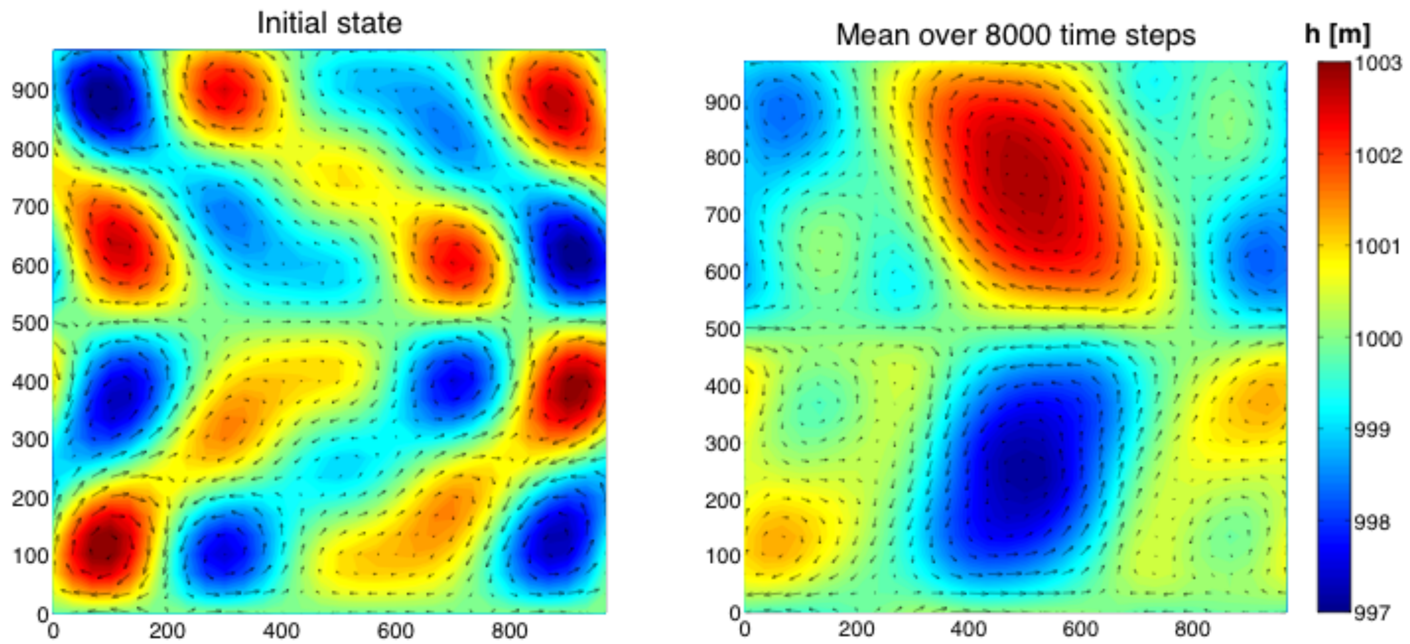


Random transformation
with constraints

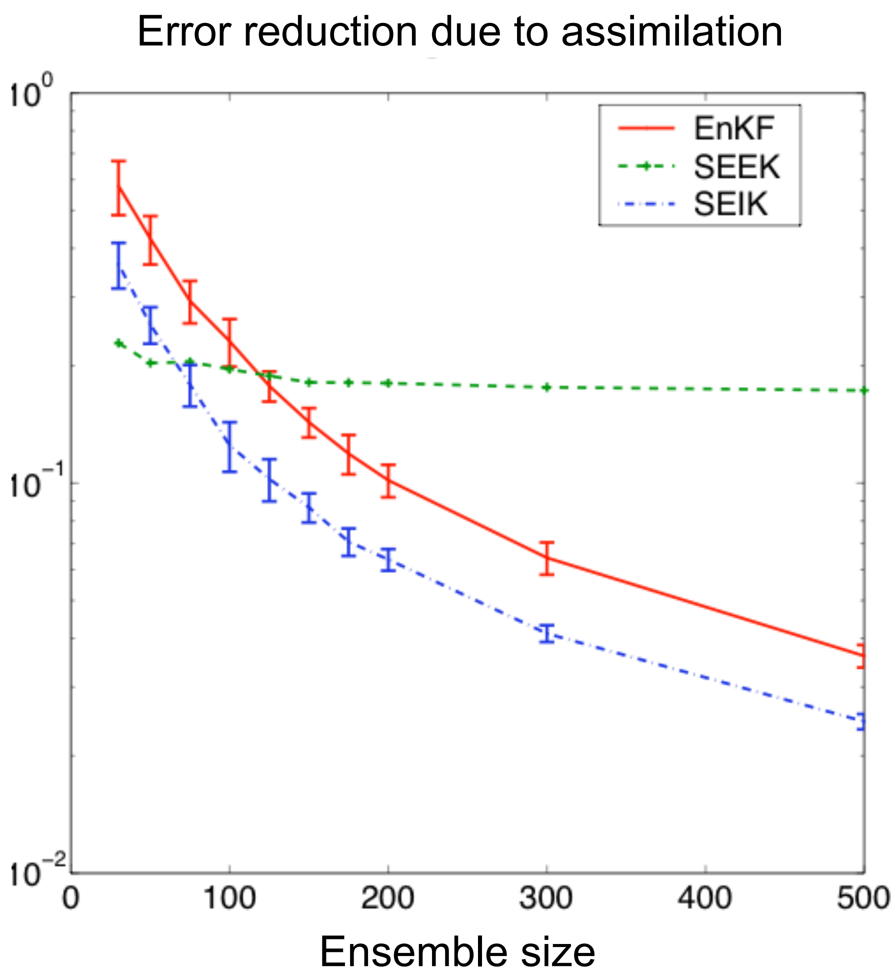
Minimum change to model states	Larger change to ensemble states
Better chance to preserve balances	More impact on balances
Preserves higher-order moments (Ensemble clustering, Amezcua et al. 2012)	Destroys higher-order moments (closer to Gaussian)

A simple test problem

- Twin experiment with nonlinear shallow water equations
- Initial state estimate: temporal mean state
- Initial cov. matrix: variability around mean state



Shallow water model: filter performances

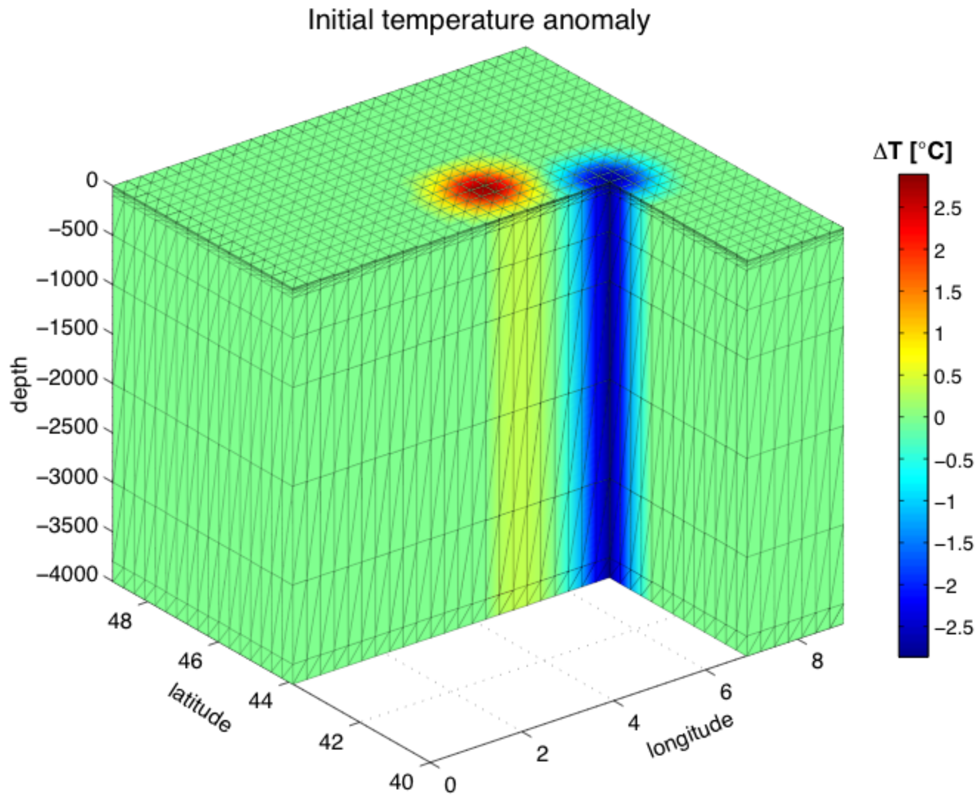


- SEEK stagnates
- same convergence behavior for EnKF and SEIK
- smaller performance for EnKF than for SEIK
- EnKF ensemble 1.5-2 times larger than SEIK ensemble for same filter performance

L. Nerger et al., Tellus 57A (2005) 715-735

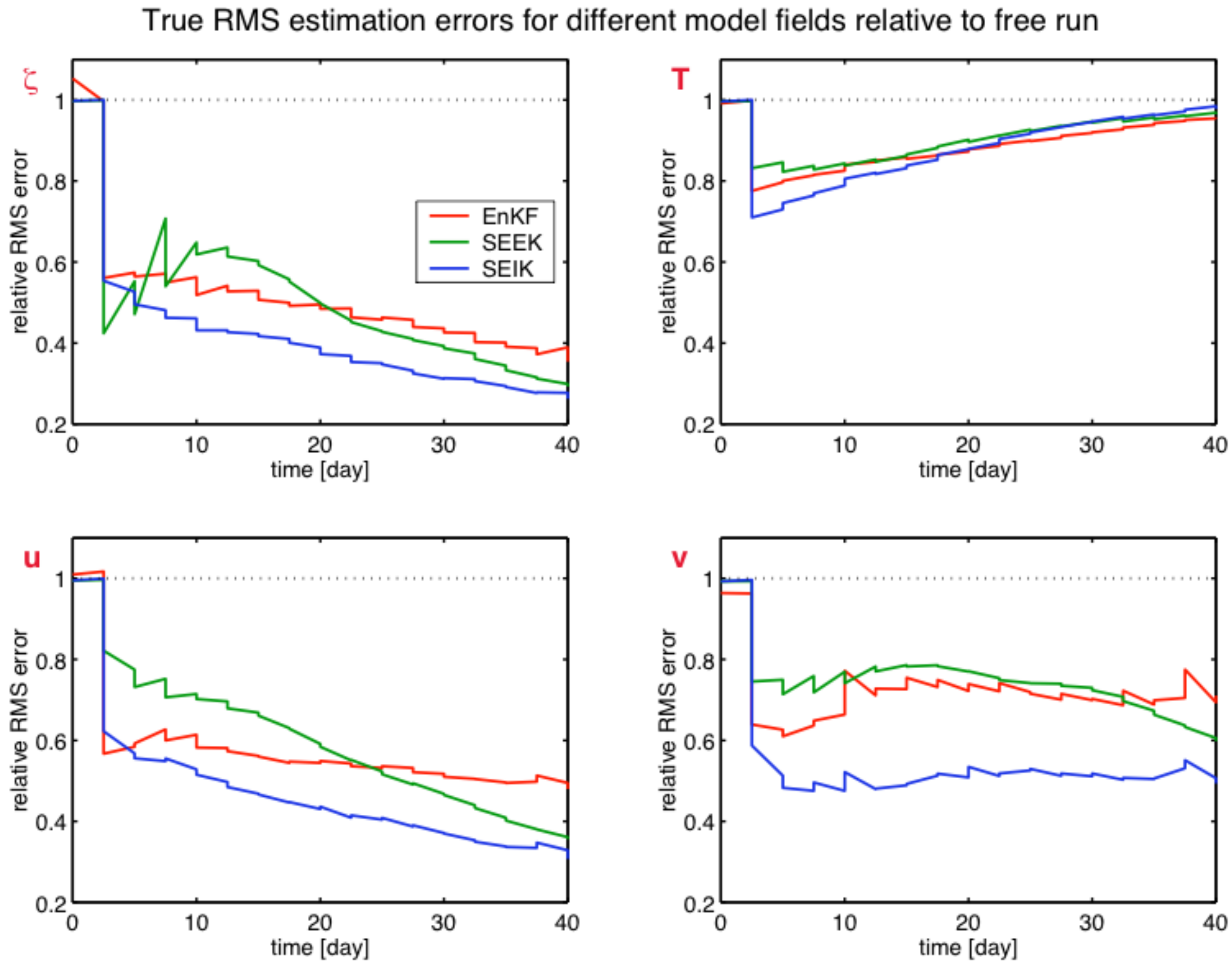
3D box experiment

- finite element model FEOM
- 31x31 grid points, 11 layers
- nonlinear problem: interacting baroclinic Rossby waves
- Assimilate sea surface height each 2.5 days over 40 days



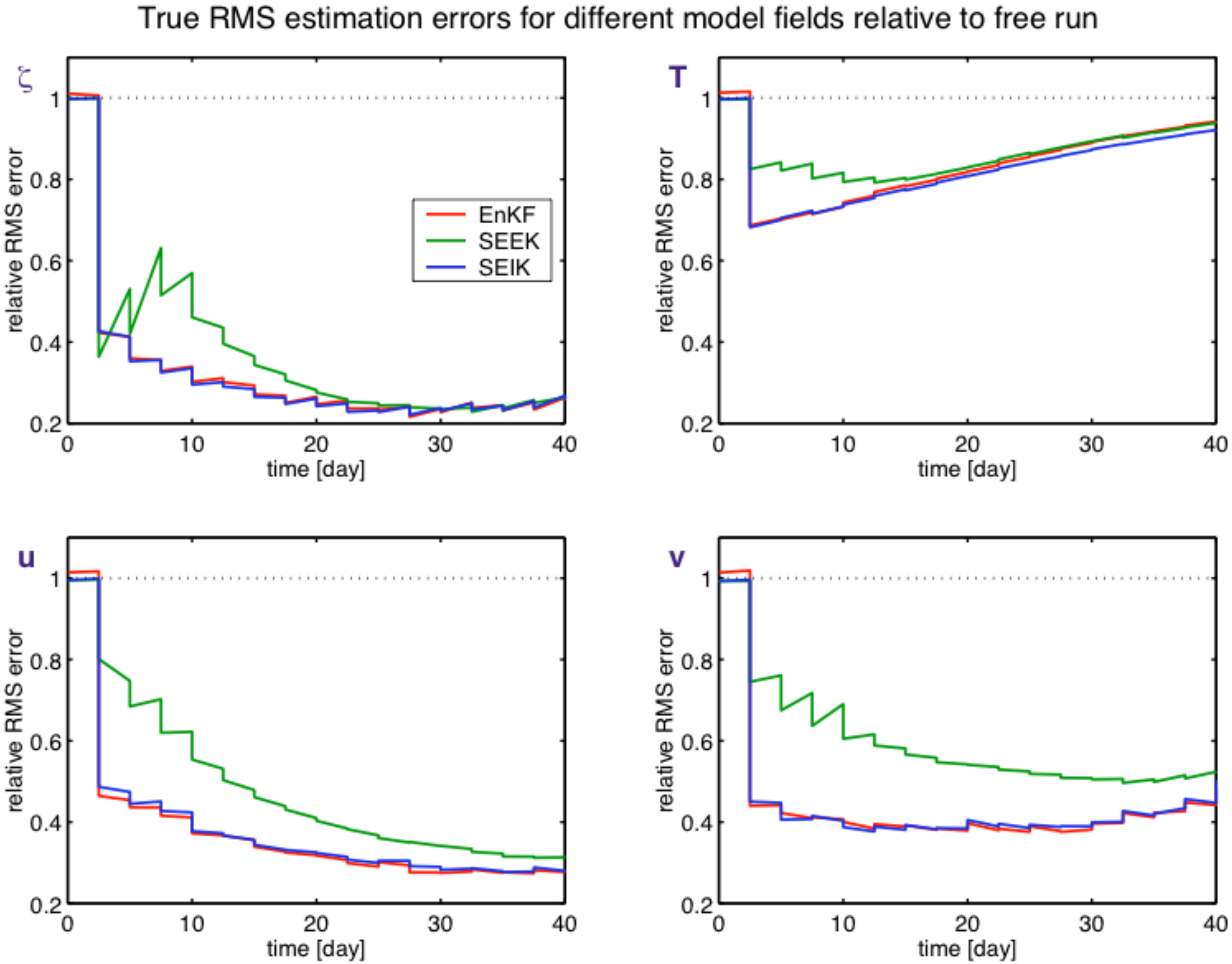
3D Box - filter performance

N=10



3D Box - filter performance

N=100



3D Box - Computation Times (N=10)

Model integrations: 6600s

Filter update:

Filter	Time
EnKF	67.8s
SEIK	0.6s

Difference due to

- inversion of large matrix in EnKF
- generation of ensemble of observations

Studying Kalman filters

- Goal: Find the assimilation method with
  - smallest estimation error
  - most accurate error estimate
  - least computational cost
  - least tuning
- Want to understand behavior, in particular performance
- Difficulty:
  - Optimality of Kalman filter well known for linear systems
  - Optimality not established for non-linear systems
  - ➜ Need to apply methods to test problems!
- One way to learn:
  - Compare different methods to learn from differences

Square-root Kalman filters

Ensemble-based/error-subspace Kalman filters

A little “zoo” (not complete):

EnKF(94/98)

Studied in Nerger et al. (2005)

RRSQRT

ROEK

SEEK

SEIK

New study (Nerger et al. 2012)

EnKF(2003)

EnKF(2004)

EAKF

EnSRF

ETKF

ESTKF

MLEF

SPKF

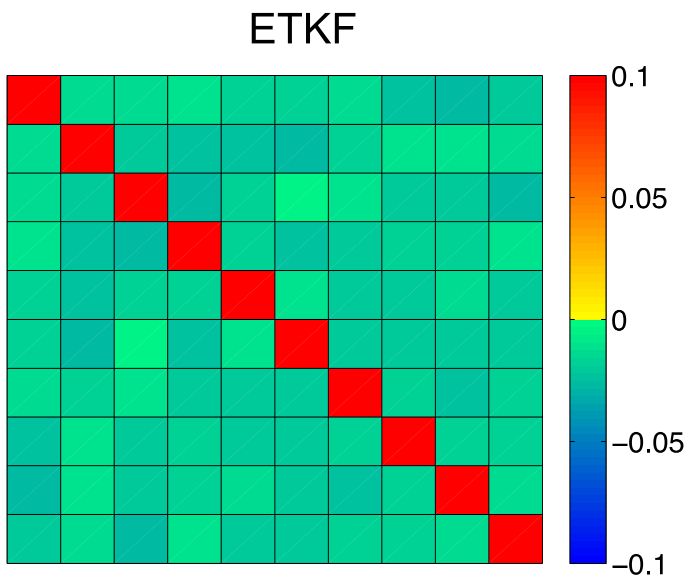
ESSE

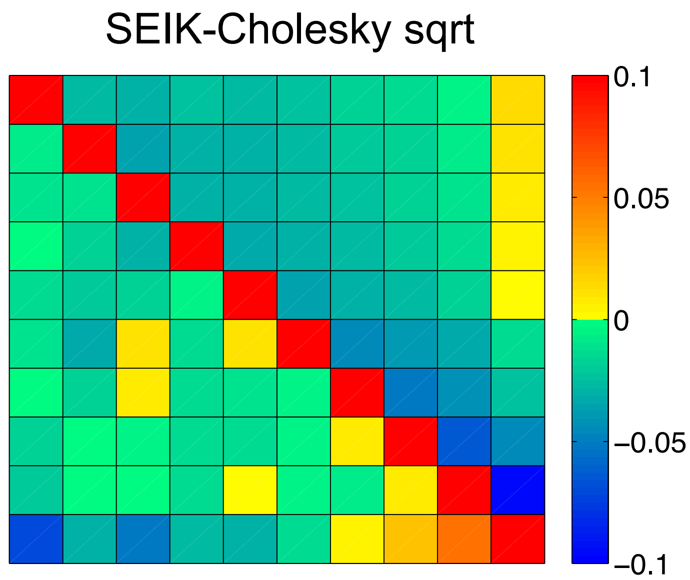
RHF

anamorphosis

New filter formulation

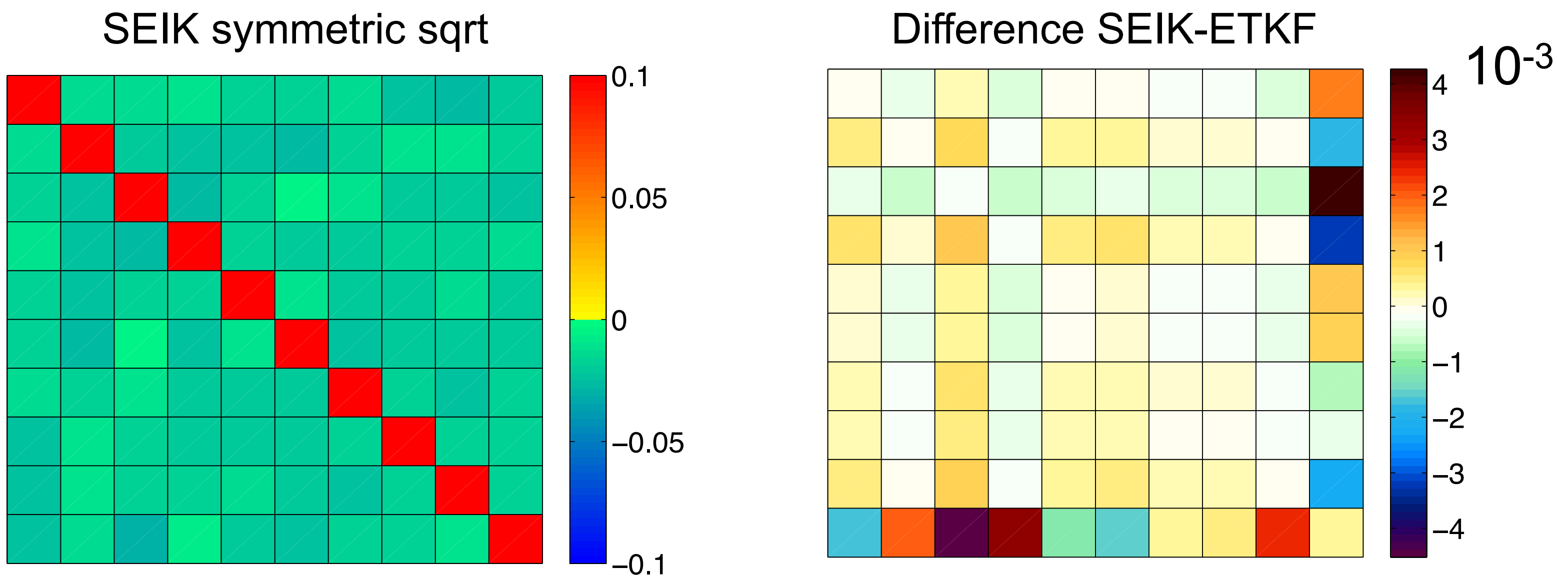
Weight Matrices (W in $X^{a'} = X^{f} W$)





ETKF	SEIK with Cholesky sqrt
main contribution from diagonal (minimum transformation)	main contribution from diagonal
Off-diagonals of similar weight	Off-diagonals with strongly varying weights
➔ Minimum change in distribution of ensemble variance	➔ Changes distribution of variance in ensemble

Transformation Matrix of SEIK/symmetric sqrt

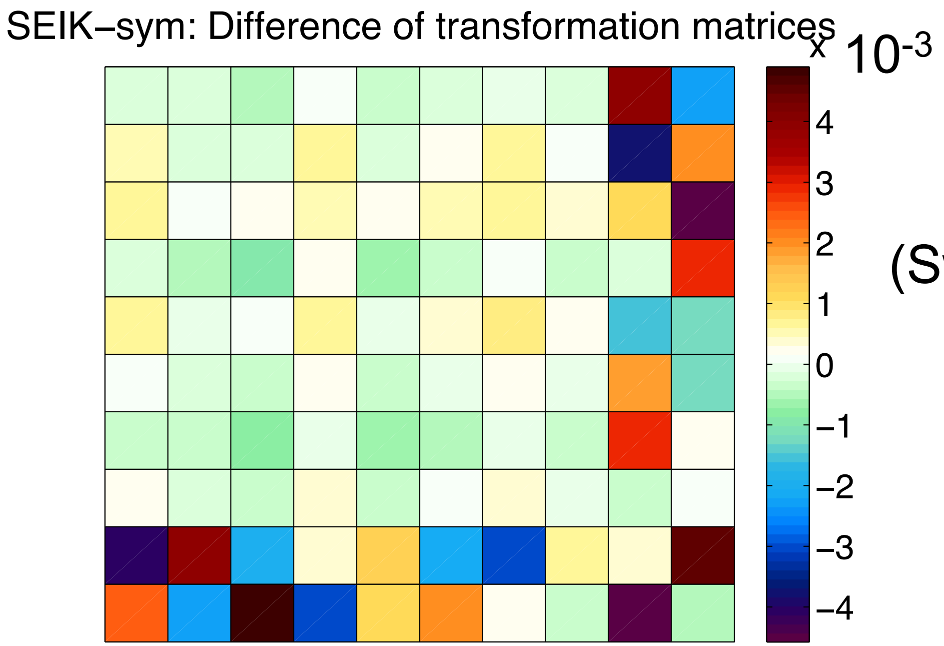


Transformation matrices of ETKF and SEIK-sym very similar

Largest difference for last ensemble member
(Experiments with Lorenz96 model: This can lead to smaller ensemble variance of this member)

SEIK depends on ensemble order

Switch last two ensemble members



(Switched back last two columns & rows for comparison)

Ensemble transformation depends on order of ensemble members
(For ETKF the difference is 10^{-15})

Statistically fine, but not desirable!

Analysis step and ensemble transformation

➜ Ensemble transformation in SEIK depends on order of ensembles

➜ Something wrong with SEIK?

Forecast Covariance: $\check{\mathbf{P}}_k^f = \mathbf{L}_k \mathbf{G} \mathbf{L}_k^T$

with
$$\mathbf{L}_k := \mathbf{X}_k^f \mathbf{T}$$
$$\mathbf{G} := \frac{1}{N-1}\left(\mathbf{T}^T\mathbf{T}\right)^{-1}$$
$$\mathbf{T}_{i,j} = \begin{cases} 1 - \frac{1}{N} & \text{for } i = j, i < N \\ -\frac{1}{N} & \text{for } i \neq j, i < N \\ -\frac{1}{N} & \text{for } i = N \end{cases}$$

➜ Matrix $\mathbf{T}$ subtracts ensemble mean and removes last column

➜ Last column depends on ensemble ordering!

Ensemble order matters in SEIK

Distinct matrices **L** ➜ distinct matrices **U**:

$$\mathbf{U}_k^{-1} = \rho\mathbf{G}^{-1} + (\mathbf{H}_k\mathbf{L}_k)^T\mathbf{R}_k^{-1}\mathbf{H}_k\mathbf{L}_k$$

$$\check{\mathbf{P}}_k^a = \mathbf{L}_k\mathbf{U}_k\mathbf{L}_k^T \quad \text{(this is always correct)}$$

➜ Finally: slightly different eigenvalues and eigenvectors

Ensemble-transformation:

Square-root $\mathbf{C}_k^{-1}(\mathbf{C}_k^{-1})^T = \mathbf{U}_k^{-1}$ (SVD)

New ensemble: $\mathbf{X}_k^a = \mathbf{X}_k^a + \sqrt{N-1}\,\mathbf{L}_k\mathbf{C}_k^T\mathbf{\Omega}_k^T$

$\mathbf{\Omega}$ is projection from N-1 to N
(Random matrix from Householder reflections)

Revised T matrix

Identical transformations require different projection matrix for SEIK:

$$\mathbf{L} \quad := \quad \mathbf{X}^f \ \mathbf{T}$$

For SEIK:

$\mathbf{T}$ subtracts ensemble mean and drops last column

➜ Dependence on order of ensemble members!

➜ Solution:

- ➜ Redefine $\mathbf{T}$: Distribute last member over first N-1 columns
- ➜ Also replace $\mathbf{\Omega}$ by new $\hat{\mathbf{T}}$

New filter formulation:

Error Subspace Transform Kalman Filter (ESTKF)

T-matrix in ESTKF

Redefine **T**:

- Subtract ensemble mean
- Distribute last column over first N-1 columns
- Use correct scaling to preserve mean

$$\hat{\mathbf{T}}_{i,j} = \begin{cases} 1 - \frac{1}{N}\frac{1}{\frac{1}{\sqrt{N}}+1} & \text{for } i = j, i < N \\ -\frac{1}{N}\frac{1}{\frac{1}{\sqrt{N}}+1} & \text{for } i \neq j, i < N \\ -\frac{1}{\sqrt{N}} & \text{for } i = N \end{cases}$$

➜ A deterministic form of $\mathbf{\Omega}$ (Householder reflection)

With this:

$$\mathbf{G} := \frac{1}{N-1}\mathbf{I}$$

New filter - ESTKF

Use redefined **T** (= deterministic $\boldsymbol{\Omega}$)

Forecast Covariance: $\check{\mathbf{P}}_k^f = \mathbf{L}_k \mathbf{G} \mathbf{L}_k^T$

With $\mathbf{L}_k := \mathbf{X}_k^f \hat{\mathbf{T}}$

Matrix **U** simplifies to:

$$\mathbf{U}_k^{-1} = \rho(N-1)\mathbf{I} + (\mathbf{H}_k \mathbf{L}_k)^T \mathbf{R}_k^{-1} \mathbf{H}_k \mathbf{L}_k$$

(inverse of error covariance matrix in error space)

Ensemble transformation

$$\mathbf{X}_k^a = \overline{\mathbf{X}}_k^a + \sqrt{N-1}\mathbf{X}_k^f \boxed{\hat{\mathbf{T}} \mathbf{C}_k^T \hat{\mathbf{T}}^T}$$

- ➜ Consistent projections between state space and error space
- ➜ Transformation identical to ETKF (same eigenvalues/vectors)
- ➜ Cheaper than ETKF
- ➜ Not more expensive than SEIK

T-matrix in SEIK and ESTKF

SEIK:
$$\mathbf{T}_{i,j} = \begin{cases} 1 - \frac{1}{N} & \text{for } i = j, i < N \\ -\frac{1}{N} & \text{for } i \neq j, i < N \\ -\frac{1}{N} & \text{for } i = N \end{cases}$$

ESTKF:
$$\hat{\mathbf{T}}_{i,j} = \begin{cases} 1 - \frac{1}{N}\frac{1}{\frac{1}{\sqrt{N}}+1} & \text{for } i = j, i < N \\ -\frac{1}{N}\frac{1}{\frac{1}{\sqrt{N}}+1} & \text{for } i \neq j, i < N \\ -\frac{1}{\sqrt{N}} & \text{for } i = N \end{cases}$$

- Efficient implementation as subtraction of means & last column
- ETKF: improve compute performance using a matrix **T**

ESTKF: New filter with identical transformation as ETKF

New filter ESTKF:

- ➜ Consistent projections between state space and error space
- ➜ Minimum Transformation identical to ETKF (or LETKF) (same eigenvalues/vectors)
- ➜ Slightly cheaper than ETKF (because of computations in N-1)
- ➜ Not more expensive than SEIK
- ➜ Transformation independent of ensemble order
- ➜ Direct access to error subspace
- ➜ smaller condition number of transform matrix **A** (**U** in ESTKF)

L. Nerger et al., Monthly Weather Review 140 (2012) 2335-2345

Nonlinearity

and current developments

Data Assimilation – an estimation problem

Probability densities: $p\left(\mathbf{x}_i\right),\ p\left(\mathbf{y}_i\right)$

Likelihood of $\mathbf{y}$ given $\mathbf{x}$: $p\left(\mathbf{y}_i|\mathbf{x}_i\right)$

Bayes law: Probability density of $\mathbf{x}$ given $\mathbf{y}$

$$p\left(\mathbf{x}_i|\mathbf{y}_i\right) = \frac{p\left(\mathbf{y}_i|\mathbf{x}_i\right) p\left(\mathbf{x}_i\right)}{p\left(\mathbf{y}_i\right)}$$

Solution of the full problem is principally known

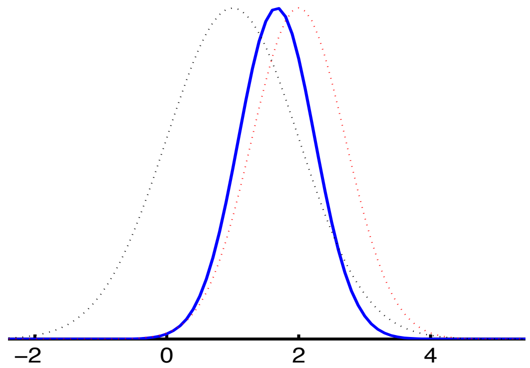
1. Time evolution of $p\left(\mathbf{x}_i\right)$ given by Fokker-Planck (forward Kolmogorov) equation
2. Apply Bayes law at time instance or interval

- This is too costly (if you don't have a tiny model)
- We don't even know the initial error distributions

Data Assimilation – Probabilistic Assumptions

Assume Gaussian distributions:

$$\mathcal{N}\left(\mu, \sigma^2\right) = a\ e^{\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)}$$



Observations: $\mathcal{N}(\mathbf{y}, \mathbf{R})$

State: $\mathcal{N}(\mathbf{x}, \mathbf{P})$

Posterior state distribution

$$p(\mathbf{x}_i | \mathbf{Y}_i) \sim a e^{-J(\mathbf{x})}$$

With

$$J(\mathbf{x}) = (\mathbf{x} - \mathbf{x}^b)^T \mathbf{P}^{-1} (\mathbf{x} - \mathbf{x}^b) + (\mathbf{y} - H[\mathbf{x}])^T \mathbf{R}^{-1} (\mathbf{y} - H[\mathbf{x}])$$

Mean state and variance fully describe the solution

Kalman Filter (Kalman, 1960)

Forecast:

State propagation

$$\mathbf{x}_i = \mathbf{M}_{i-1,i}\mathbf{x}_{i-1} + \epsilon_i$$

Propagation of error estimate

$$\mathbf{P}_i^f = \mathbf{M}_{i-1,i}\mathbf{P}_{i-1}^a(\mathbf{M}_{i-1,i})^T + \mathbf{Q}_{i-1}$$

Analysis at time t_k:

This assumes Gaussian errors of state, model, and observations!

State update

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k\left(\mathbf{y}_k - \mathbf{H}_k\mathbf{x}_k^f\right)$$

Update of error estimate

$$\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\,\mathbf{P}_k^f$$

with “Kalman gain”

$$\mathbf{K}_k = \mathbf{P}_k^f\mathbf{H}_k^T\left(\mathbf{H}_k\mathbf{P}_k^f\mathbf{H}_k^T + \mathbf{R}_k\right)^{-1}$$

Variational Data Assimilation

- Method: 4D-Var

  1. Formulate "cost function" (least squares)

$$J(\mathbf{x}_0) = \sum_{i=1}^{k} \underbrace{\left(\mathbf{x}_i - \mathbf{x}_i^b\right)^T \mathbf{C} \left(\mathbf{x}_i - \mathbf{x}_i^b\right)}_{\text{Background}} + \underbrace{\left(\mathbf{y}_i - H\mathbf{x}_i\right)^T \mathbf{D} \left(\mathbf{y}_i - H\mathbf{x}_i\right)}_{\text{Observation}}$$

  2. Minimize cost by varying $\mathbf{x}_0$ (initial state)

With **linear** model:

- $dJ/d\mathbf{x}_0$ linear function of $\mathbf{x}_0$ (theoretically solvable in one step)

With **nonlinear** model:

- $dJ/d\mathbf{x}_0$ no longer a linear function of $\mathbf{x}_0$!
- minimization might need many iterations
- Result is different from Kalman filter

Optimality of the Kalman Filter

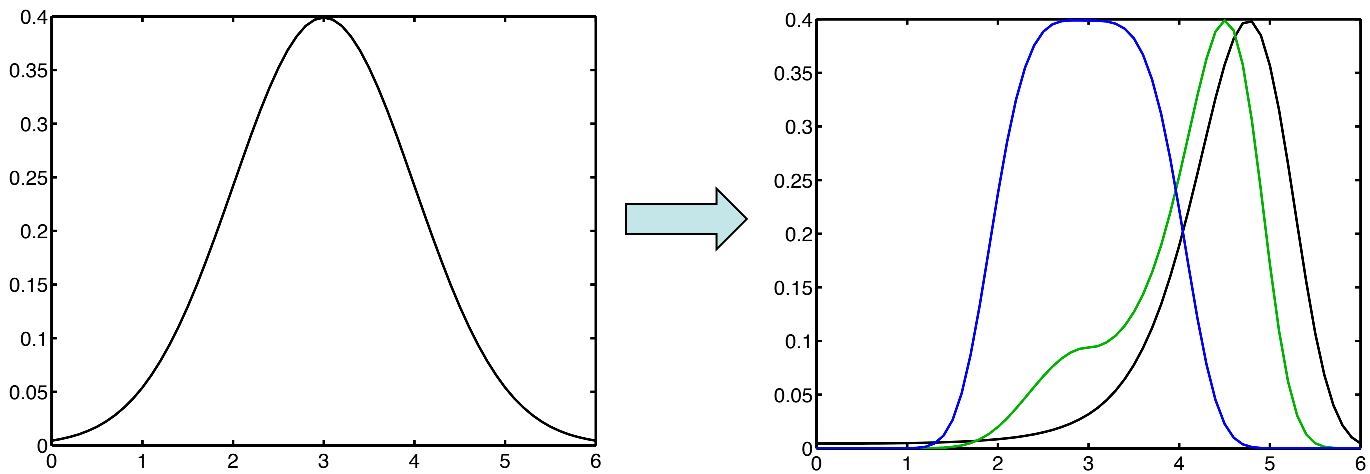
Kalman filter was derived to minimize variance

Kalman filter is optimal only if

- Covariance matrices are known
  (they are not in high-dimensional systems)
- Errors have normal distribution

With a nonlinear model

- Initial Gaussianity not preserved by nonlinear transformation

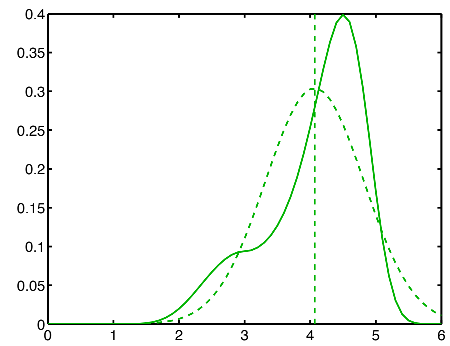


EnKF: Effect of non-Gaussian distributions

Ensemble estimates:

Mean

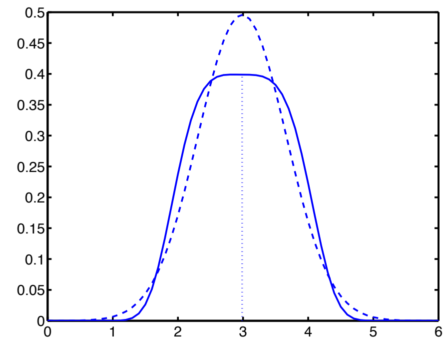
- biased if distribution is skewed
- not at maximum of distribution



➜ Biased analysis estimate

Error variance

- not a sufficient estimate of error (if used alone)
- over- or underestimates width of distribution



➜ Too big or too small state correction

➜ Sub-optimal corrections in analysis step

➜ Nonetheless:

- EnKFs work successfully well in most cases
- Compares well to 4D-Var (e.g. Buehner et al. 2005)

Some recent methods to handle non-Gaussianity

Gaussian Anamorphosis (Bertino et al. 2003)

- Transform $\mathbf{X}_k^f$ into approx. Gaussian distribution
- Used in several studies, e.g. in biogeochemistry (Simon/Bertino 2009, Doron et al. 2011)
- Gaussianity of cross-covariances might be problematic

Rank histogram filter (Anderson 2010)

- Use a rank histogram to weight ensemble members for their departure from prescribed Gaussian

Hybrid Ensemble-Variational DA

- Motivation – if you already run a 4D-Var system:
  - Stick to 4D-Var
  - Improve it by combination with ensembles

Cost function

$$J(\mathbf{x}_0) = \sum_{i=1}^{k} \underbrace{\left(\mathbf{x}_i - \mathbf{x}_i^b\right)^T \mathbf{C} \left(\mathbf{x}_i - \mathbf{x}_i^b\right)}_{\text{Background}} + \underbrace{\left(\mathbf{y}_i - H\mathbf{x}_i\right)^T \mathbf{D} \left(\mathbf{y}_i - H\mathbf{x}_i\right)}_{\text{Observation}}$$

Now, use ensemble estimate:

$$\mathbf{C}^{-1} = \tilde{\mathbf{P}}_i^f$$

- Time – and flow – dependent
- Ensemble can also help avoiding adjoint model (e.g. Liu et al. 2008)
- Low rank of C: Localization likely required (e.g. Buehner et al. 2010)

Alternative uses of Bayes law

Bayes law: Probability density of $\mathbf{x}$ given $\mathbf{y}$

$$p\left(\mathbf{x}_i|\mathbf{y}_i\right) = \frac{p\left(\mathbf{y}_i|\mathbf{x}_i\right)p\left(\mathbf{x}_i\right)}{p\left(\mathbf{y}_i\right)}$$

Represent $p(\mathbf{x}_i)$ by ensemble: $p(\mathbf{x}_i) = \frac{1}{N}\sum_{j=1}^{N}\delta(\mathbf{x}_i - \mathbf{x}_i^{(j)})$

$$p(\mathbf{x}_i|\mathbf{y}_i) = \sum_{j=1}^{N}\delta(\mathbf{x}_i - \mathbf{x}_i^{(j)})\,\frac{p(\mathbf{y}_i|\mathbf{x}_i^{(j)})}{p(\mathbf{y}_i)}$$

Kalman filter:

assume normal distributions
compute new ensemble states

$$\mathbf{x}_i^{a(j)}; j = 1, \ldots, N$$

Alternative:

keep ensemble states with weights

$$w^{(j)} = \frac{p(\mathbf{y}_i|\mathbf{x}_i^{(j)})}{p(\mathbf{y}_i)}$$

Ensemble weights – Particle Filter

Analysis probability density

$$p(\mathbf{x}_i|\mathbf{y}_i) = \sum_{j=1}^{N} \delta(\mathbf{x}_i - \mathbf{x}_i^{(j)})\, w^{(j)}$$

Computation of weights: $w^{(j)} = \dfrac{p(\mathbf{y}_i|\mathbf{x}_i^{(j)})}{p(\mathbf{y}_i)}$

$p(\mathbf{y}_i)$: Normalization constant (sum of weights = 1)

$p(\mathbf{y}_i|\mathbf{x}_i^{(j)})$: Likelihood of observations given state

Typical assumption: Gaussian observation errors

$$p(\mathbf{y}_i|\mathbf{x}_i^{(j)}) = A\ exp\left(-\frac{1}{2}\left(\mathbf{y}_i - H\mathbf{x}_i^{(j)}\right)^T \mathbf{R}^{-1}\left(\mathbf{y}_i - H\mathbf{x}_i^{(j)}\right)\right)$$

(A single number for a single particle *j*)

Not an inverse problem any more, but an estimation problem

Particle Filter (PF)

Provides analysis probability distribution as

- ensemble states (particles)
- associated weights

No assumption of Gaussian errors for model state!

Issues:

Small systems

- Many particles have low weight
  - ➜ large ensemble
  - ➜ resampling for uniform weights (e.g. Gordon et al. 1993)

High-dimensional systems

- Almost all particles have low weight
  - ➜ PF with proposal density (van Leeuwen 2009, 2010)
  - ➜ Implicit particle filter (Chorin & Tu 2009)

  Currently an active research area

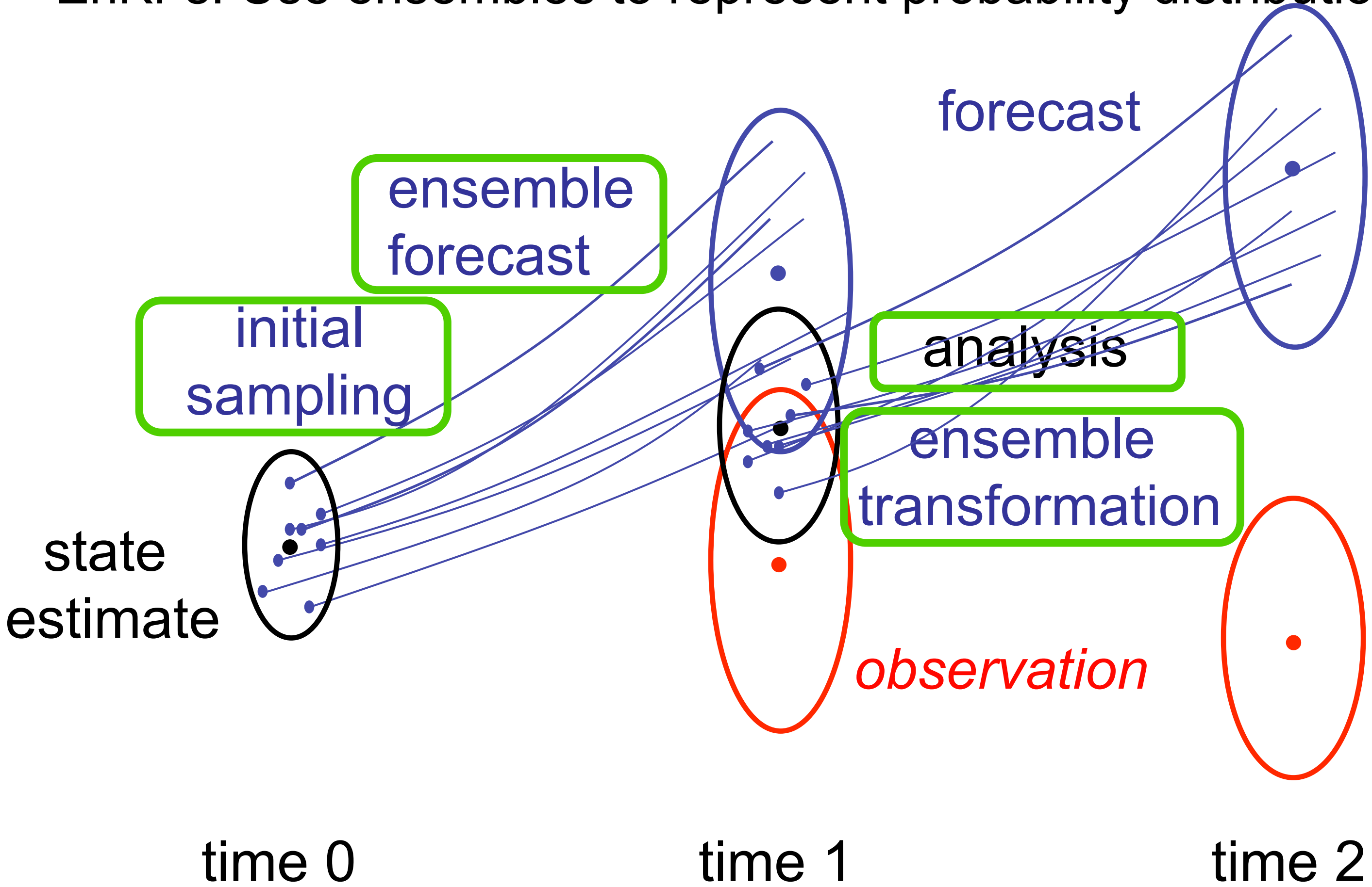
Review

Ensemble-based Kalman Filters

First formulated by G. Evensen (EnKF, 1994)

Kalman filter: express probability distributions by mean and covariance matrix

EnKFs: Use ensembles to represent probability distributions



Looks simple!

BUT:

There are many possible choices!

What we are looking for…

- Goal: Find the assimilation method with
  - smallest estimation error
  - most accurate error estimate
  - least computational cost
  - least tuning
- Want to understand and improve performance (There is no sound mathematical basis yet)
- Difficulty:
  - Optimality of Kalman filter well known for linear systems
  - No optimality for non-linear systems

  ➜ limited analytical possibilities

  ➜ apply methods to test problems

Outlook – practical aspects

Data assimilation with ensemble-based Kalman filters is costly!

Memory: Huge amount of memory required
(model fields and ensemble matrix)

Computing: Huge requirement of computing time
(ensemble integrations)

Parallelism: Natural parallelism of ensemble integration exists
(needs to be implemented)

„Fixes“: Filter algorithms do not work in their pure form
(„fixes“ and tuning are needed)
because Kalman filter optimal only in linear case

+ case studies

Thank you!