Data Assimilation for Numerical Weather Prediction

[NWP] Project: **LETKF, Efficient DA Filtering**

Diego Calderon, Maria Cesarini, Chris DeFiglia, Shuchi Goyal, Danielle Sykes, Qi Zhan, Mingjun Zhou, and Ahmed Attia

University of Arkansas
Colgate University
UNC Chapel Hill
University of Georgia
UM Baltimore County
St. John's College
University of Arizona
SAMSI/NCSU

SAMSI/NCSU UG-Workshop
November 2, 2017

Outline:

Motivation: Forecasting, DA, and Filtering

- Filtering
- EnKF, and LETKF

Process

Numerical Results: LETKF with a QG model

- Accuracy: RMSE
- Efficiency: CPU-Time

Conclusion

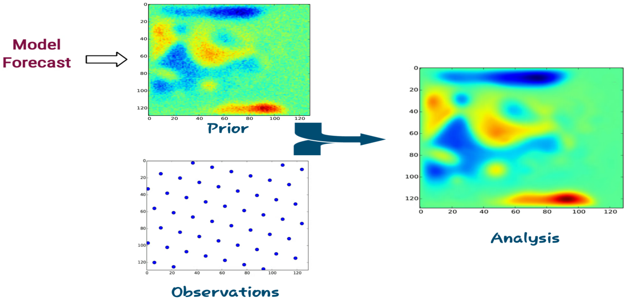
Further Implications

Acknowledgements

Motivation: Forecasting, Data Assimilation, and Filtering

- **What is Data Assimilation (DA)?**
  - It is a fusion of information collected from different sources, including model forecasts and observations in order to obtain a more accurate prediction
  - Example: ETA prediction on a GPS
- **Using DA to forecast weather:**
  - Combined with a forward forecasting model, DA can be used to predict the state of the atmosphere in the future, e.g. for the following day.

- **Assimilation**:



$$\underbrace{\textbf{Model + Prior + Observations}}_{\text{with associated \underline{uncertainties}}} \rightarrow \underbrace{\textbf{Best description of the state}}_{e.g. \textit{Filtering}}$$

- **Applications include**: Object tracking (e.g. ETA for GPS tracking), atmospheric forecasting, power flow, oil reservoir, volcano simulation, etc.

Filering:

- **Filtering** refers to the process of assimilating a single observation (vector) at a time.
- A filtering step is carried out by applying a prediction/correction procedure.
- The prediction is carried out by forwarding propagation of model dynamics.
- The correction step updates the forecasted state along with the associated uncertainty given the observation.
- The filtering process is applied sequentially in operational settings.

> The most popular/operational filter for DA applications is EnKF; EnKF/DEnKF is an ensemble-based approximation of Kalman Filter.

Project Goal

Using DATeS:

- Implement two flavors of LETKF filtering algorithm; local, and global LETKF,
- Test and analyze the accuracy and efficiency of localized LETKF filter [1] against global LETKF [2] and DEnKF Implementation [3].

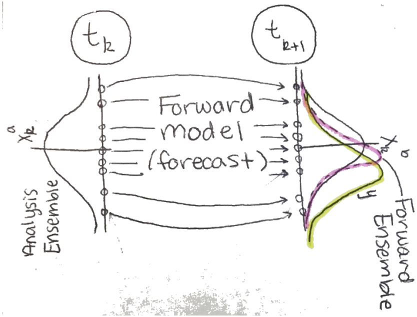
[1] Harlim, John, and Brian R. Hunt. *"Local Ensemble Transform Kalman Filter: An Efficient Scheme for Assimilating Atmospheric Data."* preprint (2005).

[2] Attia, Ahmed, and Adrian Sandu. "DATeS: A Highly-Extensible Data Assimilation Testing Suite." arXiv preprint arXiv:1704.05594 (2017).

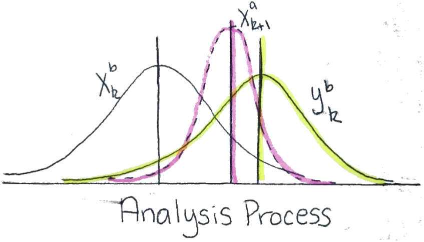
LETKF Filter

- **What is LETKF?**
  - **EnKF** stands for **En**semble **K**alman **F**ilter
  - EnKF:
    1. takes a forecast ensemble (that approximates prior distribution) and an observation
    2. uses the forecasted ensemble, along with the observation, to calculate a posterior ensemble
    3. the posterior (corrected) ensemble approximates the actual distribution
  - **LETKF** encompasses the uncertainty of each estimate based on its local region

Filtering Cycles: Forecast & Assimilation/Analysis



(a)



(b)

Localization in LETKF

- **Global versus Local LETKF:**
  - The original/Global formulation of the filter results in spurious correlations,
  - It is inefficient to assimilate observations from the whole domain
  - Local models only use information close to a certain point to make predictions about that point at a future state

Localization in LETKF (cont'd):

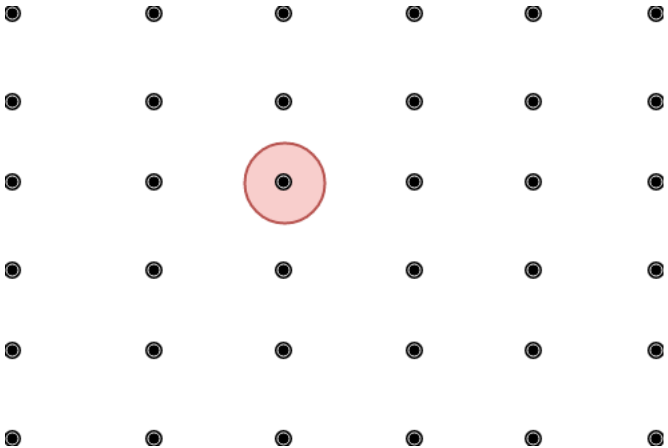


Figure: A visualization of localized LETKF

Project Goal

Using DATeS, implement, test and analyze the accuracy and efficiency of localized LETKF filter against global LETKF and DEnKF

Analysis Step

Algorithm 1 LETKF Analysis Algorithm: Global

1: **procedure** LETKF_GLOBAL
Input: A forecast ensemble ($\mathbf{X}$), and an observation $\mathbf{y}^o$
Output: An ensemble of states from the posterior distribution $\mathbf{X}^a$
2: Apply $\mathcal{H}$ to each column of $\mathbf{X}$ to get $\mathbf{Y}$. Average its columns to get the vector $\bar{\mathbf{y}}^b \in \mathbb{R}^o$ and subtract $\bar{\mathbf{y}}^b$ from each column of $\mathbf{Y}$ to get $\mathbf{Y}^b \in \mathbb{R}^{o\times k}$
3: Average the columns of $\mathbf{X}$ to get $\bar{\mathbf{x}}^b \in \mathbb{R}^s$, and subtract it from $\mathbf{X}$ to get $\mathbf{X}^b \in \mathbb{R}^{s\times k}$
4: Compute $\mathbf{C} = (\mathbf{Y}^b)^T \cdot \mathbf{R}^{-1}$, $\mathbf{C} \in \mathbb{R}^{k\times o}$
5: Compute $\tilde{\mathbf{P}}^a = [(k-1)\cdot \mathbf{I} + \mathbf{C}\mathbf{Y}^b]^{-1}$, $\mathbf{I} \in \mathbb{R}^{k\times k}$
6: Compute $\mathbf{W}^a = [(k-1)\tilde{\mathbf{P}}^a]^{\frac{1}{2}}$, $\mathbf{W}^a \in \mathbb{R}^{k\times k}$
7: Compute $\mathbf{w}^a = \tilde{\mathbf{P}}^a\mathbf{C}(\mathbf{y}^o - \bar{\mathbf{y}}^b)$, $\mathbf{w}^a \in \mathbb{R}^k$ and add it to each column of $\mathbf{W}^a$ to get $\mathbf{W} \in \mathbb{R}^{k\times k}$
8: Compute $\mathbf{X}^b\mathbf{W}$ and add $\bar{\mathbf{x}}^b$ to each column
9: **end procedure**

Analysis Step

Algorithm 2 LETKF Analysis Algorithm: Local

procedure LETKF_LOCAL
Input: A forecast ensemble ($\mathbf{X}$), and an observation $\mathbf{y}^o$
Output: An ensemble of states from the posterior distribution $\mathbf{X}^a$
  Repeat steps 2, and 3 in Algorithm 1
  for `<each model grid-point>` **do**
    Truncate $\bar{\mathbf{x}}^b$, and $\mathbf{X}^b$ to include only the model variables at that grid point.
    Truncate $\mathbf{y}^o$, $\bar{\mathbf{y}}^b$, and $\mathbf{Y}^b$ to include only the observations within radius r around that grid point.
    Repeat steps $4 - 7$ from Algorithm 1 given the truncated matrices
    Use the calculated update, to calculate the analysis at the current grid-point
  end for
end procedure

DEnKF Filter Accuracy: RMSE

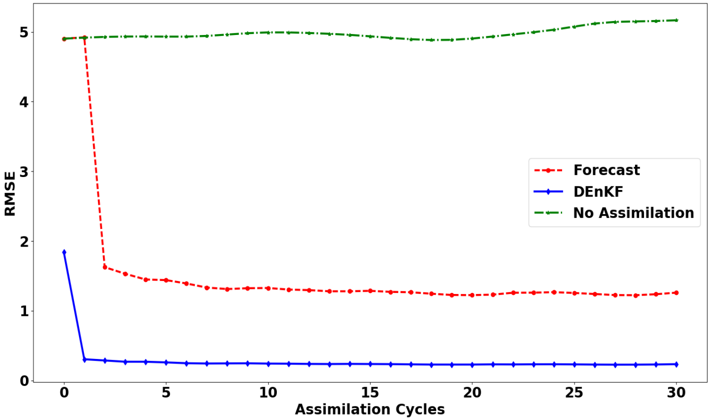
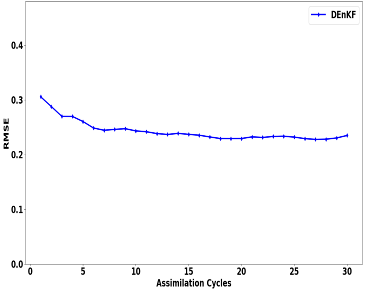
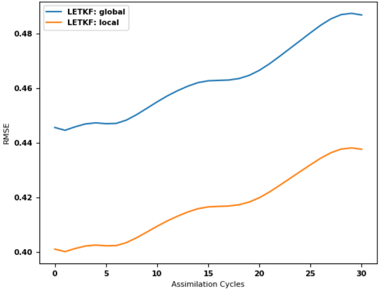


Figure: Root Mean Square Error for DEnKF (Benchmark). The Error is calculated as the difference between the analysis state (ensemble-mean), and the true/reference solution.

Filters' Accuracy: RMSE



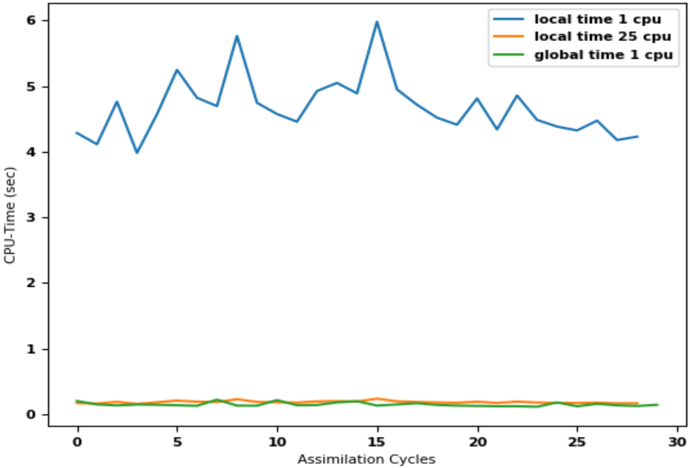
(a) RMSE: DEnKF



(b) RMSE: LETKF

Figure: Root Mean Square Error for DEnKF and LETKF. The Error is calculated as the difference between the analysis state (ensemble-mean), and the true/reference solution.

CPU-Time Comparison



Conclusion

- **We have gained professional experience about:**
  1. Gridded models, prediction, inverse problems, and data assimilation,
  2. Advanced Python skills (e.g. Numpy, Scipy, Matplotlib, Python inheritance & classes, etc.),
  3. DATeS package for data assimilation
- **We were able to implement two flavors of the LETKF filter:**
  1. Global LETKF,
  2. Local LETKF.
- **We have also demonstrated the benefits of localization versus globalization, e.g. improved accuracy, and computational cost.**

Further Implications

Given more time, we would:

1. Run/Test the code we implemented for larger model settings,
2. Run LETKF in parallel,
3. Study the effect of changing the localization radius (radius of influence) on the filter performance, e.g. RMSE.

Acknowledgements:

Thank you to the following people for their contributions to this project:

- **SAMSI, and NCSU**, and **Coordinators** of the SAMSI UG Workshop
- **DATeS Team:** `http://people.cs.vt.edu/~attia/DATeS/About.html`
- **SAMSI Postdocs**
- **NWP-Project Team**
  - Diego Calderon;
  - Maria Cesarini;
  - Chris DeFiglia;
  - Shuchi Goyal;
  - Danielle Sykes;
  - Qi Zhan;
  - Mingjun Zhou;
  - Ahmed Attia; SAMSI/NCSU Postdoc.

Questions!