DA for Ensembles (EDA, LETKF)

Simon Toedtli

Slides adapted from Chris Snyder and Tao Sun

Ensemble DA

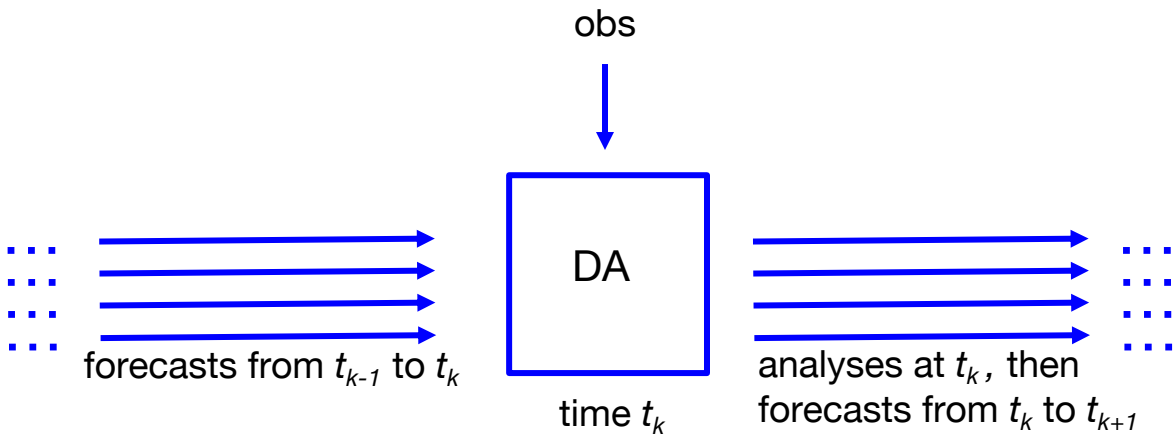
… uses an ensemble of forecasts to inform the background covariance **B**

- e.g., 3D and 4D EnVar, from yesterday

DA for Ensembles

Ensemble forecasts begin from an ensemble of initial conditions

To cycle ensemble DA, need DA that produces an ensemble of ICs



DA for Ensembles

What properties should the ensemble of ICs have?

Ideally, ensemble members would be drawn randomly from $p(\mathbf{x} \mid \mathbf{y}^o)$

In practice, must accept approximations to that condition

- Ensemble has mean, covariance consistent with the Kalman filter

DA for Ensembles

Consider two schemes (out of many) here:

- Ensemble of Data Assimilations (EDA); JEDI implementation by Gas and Tremolet
- Local Ensemble Transform Kalman Filter (LETKF); JEDI implementation by Frolov et al. (2023, *JAMES)*

Ensemble of Data Assimilations (EDA)

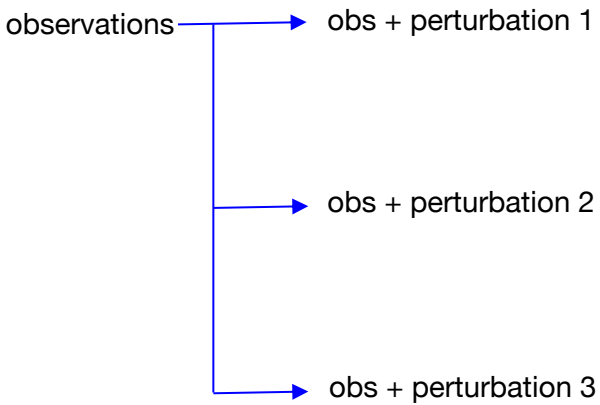
Run N_{ens} instances of chosen DA algorithm (e.g. EnVar, 3DVar). For each member:

- "perturb" obs by adding realization from $N(0, \mathbf{R})$ → asymptotically correct posterior covariance
- EnVar: member-specific $\mathbf{B}_{e,i}$ through self-exclusion → helps to produce correct posterior spread

Ensemble of Data Assimilations (EDA)

Run N_{ens} instances of chosen DA algorithm (e.g. EnVar, 3DVar). For each member:

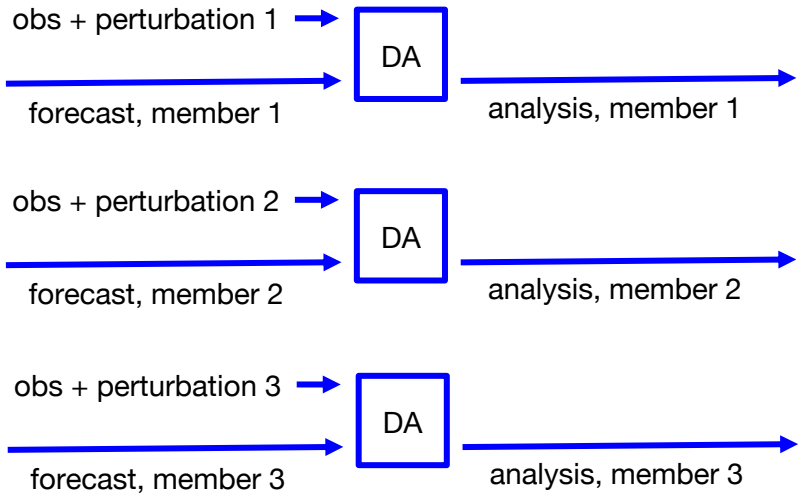
- "perturb" obs by adding realization from $N(0, \mathbf{R})$ → asymptotically correct posterior covariance
- EnVar: member-specific $\mathbf{B}_{e,i}$ through self-exclusion → helps to produce correct posterior spread



Ensemble of Data Assimilations (EDA)

Run N_{ens} instances of chosen DA algorithm (e.g. EnVar, 3DVar). For each member:

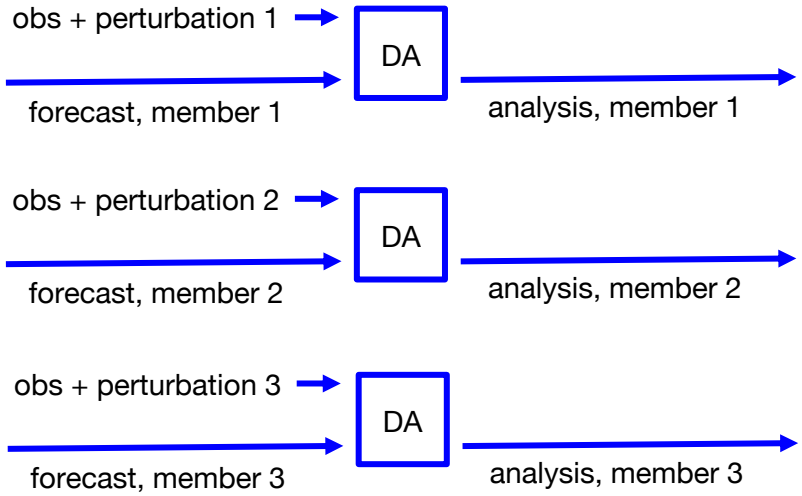
- "perturb" obs by adding realization from $N(0, \mathbf{R})$ → asymptotically correct posterior covariance
- EnVar: member-specific $\mathbf{B}_{e,i}$ through self-exclusion → helps to produce correct posterior spread



Ensemble of Data Assimilations (EDA)

Run N_{ens} instances of chosen DA algorithm (e.g. EnVar, 3DVar). For each member:

- "perturb" obs by adding realization from $N(0, \mathbf{R})$ → asymptotically correct posterior covariance
- EnVar: member-specific $\mathbf{B}_{e,i}$ through self-exclusion → helps to produce correct posterior spread



Localization implemented in model space $\mathbf{L} \circ \mathbf{B}_{e,i}$

Configure and Run JEDI EDA

Very similar to deterministic DA, additional settings to configure:

❖ Introduction of observation random errors

Set “obs perturbations” to true in the observations section:

```
observations:
  obs perturbations: true
```

For each observation type, the observation error should be

```
obs error:
  covariance model: diagonal
  zero-mean perturbations: true
  member: 1 # index of EDA member
  number of members: 10 # number of ensemble members
```

Configure and Run JEDI EDA

Very similar to deterministic DA, additional settings to configure:

- EnVar applications:
  Self-exclusion in ensemble-based background error covariance

```
cost function:
    background error:
        members from template:
            nmembers: 9 # number of ensemble members – 1
            except: [1]  # index of EDA member to exclude
```

Configure and Run JEDI EDA

Multiple EDA members can run with a single job, or all EDA members can run in parallel with multiple jobs.

For each EDA member, a specific yaml file is needed, e.g., *3denvar_${member}.yaml*.

❖ Each EDA member can be run with a single command:
${MPI_CMD} ./mpasjedi_variational.x 3denvar_${member}.yaml

❖ Or we can write all/part yaml files into the eda.yaml, and run multiple EDA members with the command:
${MPI_CMD} ./mpasjedi_eda.x eda.yaml

→ using eda.x requires more nodes, memory

eda.yaml:
files:
  - 3denvar_1.yaml
….
  - 3denvar_10.yaml

LETKF

Implements Kalman filter assuming background covariance == $\mathbf{B}_e$.

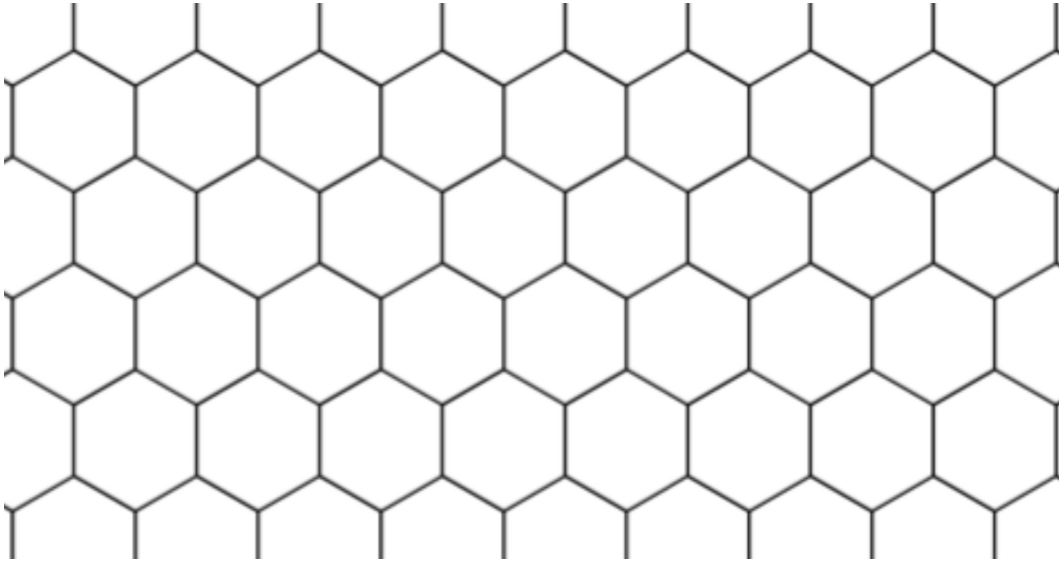
Unlike EDA,

- no minimization
- does not require TLM and adjoint of H(**x**)
- deterministic algorithm – no random numbers

LETKF

Key approximation: analysis at any point depends only on a spatially local subset of obs

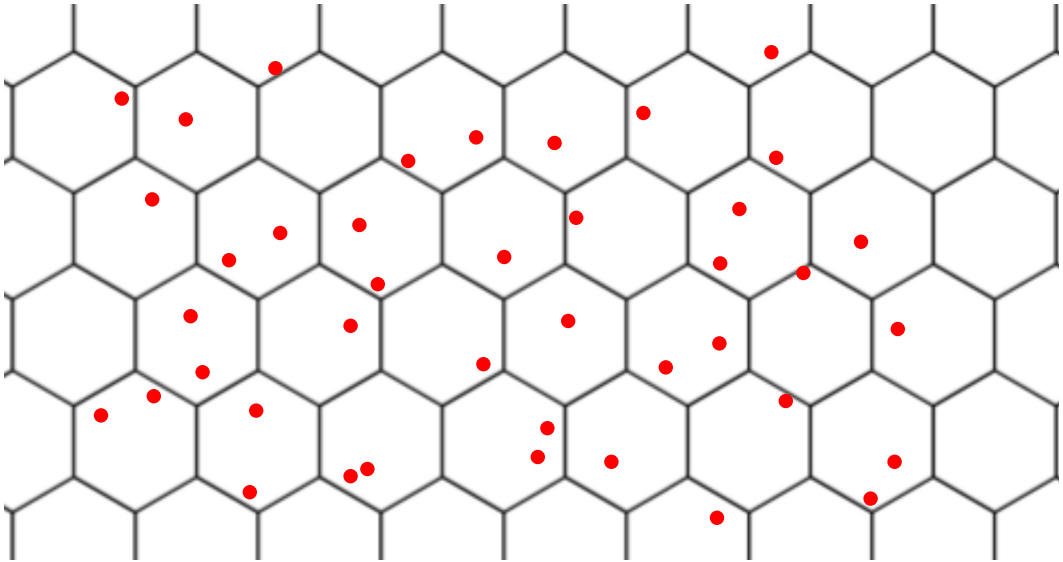
- analyses at each point can be done in parallel
- “obs space” counterpart to localization of $\mathbf{B}_e$ in EnVar
- implemented by smoothly increasing diag(**R**) with distance from analysis point



LETKF

Key approximation: analysis at any point depends only on a spatially local subset of obs

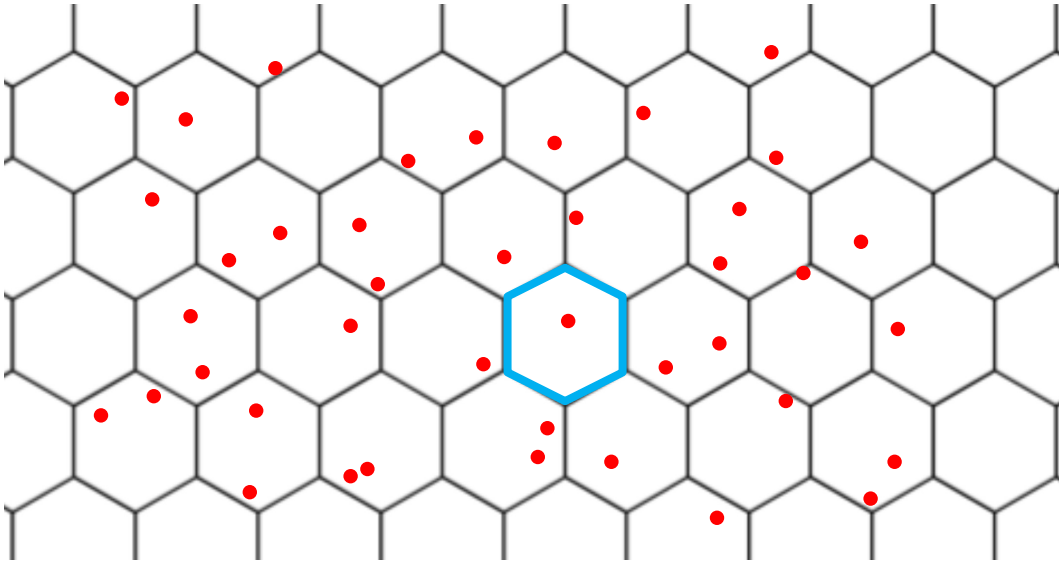
- analyses at each point can be done in parallel
- “obs space” counterpart to localization of $\mathbf{B}_e$ in EnVar
- implemented by smoothly increasing diag(**R**) with distance from analysis point



LETKF

Key approximation: analysis at any point depends only on a spatially local subset of obs

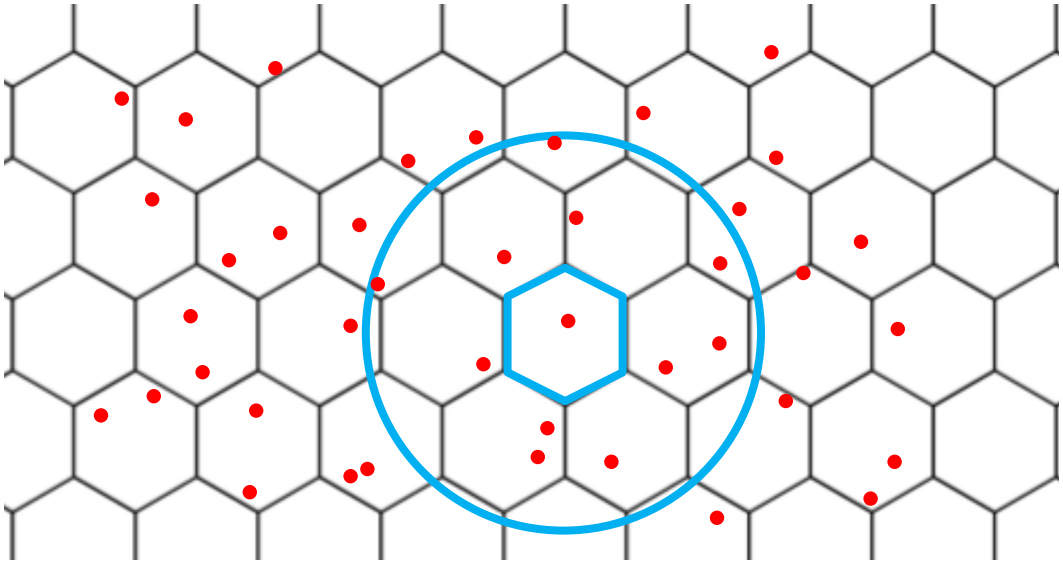
- analyses at each point can be done in parallel
- “obs space” counterpart to localization of $\mathbf{B}_e$ in EnVar
- implemented by smoothly increasing diag(**R**) with distance from analysis point



LETKF

Key approximation: analysis at any point depends only on a spatially local subset of obs

- analyses at each point can be done in parallel
- “obs space” counterpart to localization of $\mathbf{B}_e$ in EnVar
- implemented by smoothly increasing diag(**R**) with distance from analysis point



MPAS-JEDI Implementation of LETKF

"Observer": Compute H(**x**) for each forecast member

- can parallelize over members

"Solver": Update ensemble mean and ensemble deviations at each point

- uses H(x)'s from previous step
- result is an ensemble of analyses (mean + deviations)

"DiagOMA": Compute H(**x**) for each analysis, for diagnostics

Configure and Run JEDI LETKF

Yaml sections similar to variational DA, but differently organized

Configurations specific to LETKF

- Local ensemble DA: configuration of local volume solver

  LETKF: vertical localization in observation space

  GETKF: vertical localization in model space

```
local ensemble DA:
    solver: Deterministic LETKF # or Deterministic GETKF
    vertical localization: # only needed for GETKF (model-space loc)
    inflation: # more on this later
        rtps: 0.0
        rtpp: 0.8
        mult: 1.0
```

Configure and Run JEDI LETKF

Yaml sections similar to variational DA, but differently organized

Configurations specific to LETKF

- Driver: configuration of solver step (observer, solver, DiagOMA)

  Observer and DiagOMA configuration: main difference is file input and output

```
driver:
    run as observer only: true
```

Configure and Run JEDI LETKF

Yaml sections similar to variational DA, but differently organized

Configurations specific to LETKF

- Driver: configuration of solver step (observer, solver, DiagOMA)

  Solver configuration:

```
driver:
    read HX from disk: true # written by previous observer step
    save posterior ensemble: true
    save posterior mean: true
    do posterior observer: false # done by subsequent DiagOMA
```

Configure and Run JEDI LETKF

Yaml sections similar to variational DA, but differently organized

Configurations specific to LETKF

- Observations: distribution among processors

```
observations:
  observers:
  - obs space:
      distribution:
        name: RoundRobin  # RoundRobin for observer, Halo for solver
```

Configure and Run JEDI LETKF

Yaml sections similar to variational DA, but differently organized

Configurations specific to LETKF

❖ Observations: localization (only a subset of all options is shown)

```
observations:
  observers:
  - obs space:
    obs localizations:
    - localization method: Horizontal Gaspari-Cohn  # among others
      lengthscale: 1200000
      max obs: 1000
    - localization method: Vertical localization # only needed for LETKF
      vertical lengthscale: 6000.0
      ioda vertical coordinate: height
      localization function: Gaspari Cohn # among others
```

Evaluating the Quality of Ensembles for DA

Ensemble DA uses forecast ensembles to estimate variances and covariances

Can we evaluate how suitable an ensemble is for this purpose?

Available information is time series of:

- Observations
- Ensembles of forecasts mapped to the observation variables through H

Evaluating the Quality of Ensembles for DA

Fact: the expected squared observation-minus-background difference is

$$E\left((y - \mathbf{H}\mathbf{x}^b)^2\right) = \mathrm{var}(\mathbf{H}\mathbf{x}) + \sigma_o^2$$

Estimate right side from ensemble at each analysis time, then average over times,
Estimate left side from actual obs-background differences.

Equality is a necessary but not sufficient condition for a good ensemble

Underdispersion of Ensembles

Ensemble forecasts are often underdispersive; that is, their spread (= sqrt(variance)) is too small.

Addressed through various techniques

- Localization (model space, observation space, or both) in ensemble DA
- Direct modification of ensemble spread

  multiplicative inflation: multiply deviations from mean by a fixed factor > 1

  relaxation to prior perturbation: analysis deviations ← weighted average of analysis and forecast deviations

  relaxation to prior spread: multiply deviations by factor > 1, so that σ^a increases by fixed fraction of σ^b - σ^a

Inflation configuration in JEDI

EDA:

- Requires external application
- Only RTPP available

L/GETKF:

- Inflation built into local volume solvers
- Available options: RTPP, RTPS, multiplicative inflation

EDA Equations

$$J_i(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \mathbf{x}^i)^T \mathbf{B}_i^{-1}(\mathbf{x} - \mathbf{x}^i) + \frac{1}{2}(\mathbf{y}^o - \mathbf{H}\mathbf{x})^T \mathbf{R}^{-1}(\mathbf{y}^o - \mathbf{H}\mathbf{x})$$

$$\mathbf{x}^{a,i} = \text{minimizer of } J_i(\mathbf{x})$$

In tutorial example, $\mathbf{B}_i$ is the localized ensemble covariance excluding $\mathbf{x}^i$, i.e.,

$$.\mathbf{B}_i = \mathbf{L} \circ \left((N_e - 1)^{-1} \sum_{j \neq i} (\mathbf{x}^j - \hat{\mathbf{x}})(\mathbf{x}^j - \hat{\mathbf{x}})^T \right)$$

LETKF Equations

Given an ensemble of forecasts $\mathbf{x}^i$, $i = 1, \ldots, N_e$:

$$\hat{\mathbf{x}} = {N_e}^{-1} \sum_{i=1}^{N_e} \mathbf{x}^i, \qquad \delta\mathbf{x}^i = \mathbf{x}^i - \hat{\mathbf{x}}, \qquad \mathbf{X} = \text{matrix with columns } \delta\mathbf{x}^i (N_e - 1)^{-1/2}$$

$$\mathbf{y}^i = H(\mathbf{x}^i), \qquad \delta\mathbf{y}^i = \mathbf{y}^i - \hat{\mathbf{y}} \approx \mathbf{H}\delta\mathbf{x}^i, \qquad \mathbf{Y} = \text{matrix with columns } \delta\mathbf{y}^i (N_e - 1)^{-1/2}$$

$$\tilde{\mathbf{P}}^a = \left(\mathbf{I} - \mathbf{Y}^T \mathbf{R}^{-1} \mathbf{Y} \right)^{-1}, \qquad \mathbf{w}^a = \tilde{\mathbf{P}}^a \mathbf{Y}^T \mathbf{R}^{-1} (\mathbf{y}^o - \hat{\mathbf{y}}), \qquad \mathbf{W} = (\tilde{\mathbf{P}}^a)^{1/2}$$

ensemble-mean analysis: $\hat{\mathbf{x}}^a = \hat{\mathbf{x}} + \mathbf{X}\mathbf{w}^a$

analysis perturbations: $\delta\mathbf{x}_i^a =$ ith column of $\mathbf{X}\,\mathbf{W}$

Inflation Techniques

Let $\delta\mathbf{x}^i = \mathbf{x}^i - \hat{\mathbf{x}}$ be the deviation of the ith member from the ensemble mean

multiplicative prior inflation: $\delta\mathbf{x}^i \leftarrow \alpha\,\delta\mathbf{x}^i$

relaxation to prior perturbation (RTPP): $\delta\mathbf{x}^i \leftarrow (1-\alpha)\delta\mathbf{x}^{a,i} + \alpha\delta\mathbf{x}^i$

relaxation to prior spread (RTPS): $\delta\mathbf{x}^i \leftarrow (1+\alpha(\sigma_b - \sigma_a)/\sigma_a)\,\delta\mathbf{x}^i$