What to use when?

With a brief recap

Amos Lawless

Data Assimilation Research Centre & NCEO, Univ. of Reading

With thanks to Ross Bannister

Reminder: what is data assimilation?

- To blend information from models and observations.
  - State/parameter estimation (some kind of 'optimal' blending).
  - The posterior PDF or certain moments of it.

Bayes' theorem

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{x}) \times p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}$$

$$\text{posterior dist.} = \frac{\text{prior dist.} \times \text{likelihood}}{\text{normalizing constant}}$$

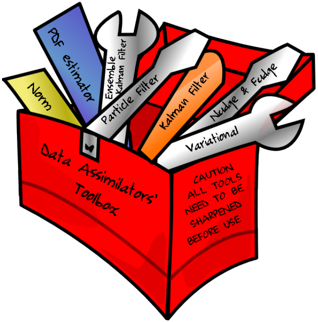
- Prior distribution: PDF of the state before observations are considered (e.g. PDF of model forecast).
- Likelihood: PDF of observations given that the state is $\mathbf{x}$.
- Posterior distribution: PDF of the state given the observations.

Confused? Overwhelmed?

In realistic practical applications we cannot represent the PDFs explicitly, so we need approximate DA methods

- Variational data assimilation
- Kalman filter (+ extended KF)
- Ensemble Kalman filters
- En-Var filters
- Hybrid methods
- Particle filters

Which method should you use for your application?



Variational data assimilation

▷ Forecast is mean of the prior, analysis is mode of the posterior (minimises a cost fn); ▷ OK when n is large;.

$$J[\mathbf{x}_0] = \frac{1}{2}(\mathbf{x}_0 - \mathbf{x}^b)^{\mathrm{T}}\mathbf{B}_0^{-1}(\mathbf{x}_0 - \mathbf{x}^b) + \frac{1}{2}\sum_{t=0}^{T}(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}))^{\mathrm{T}}\mathbf{R}_t^{-1}(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}))$$

$$\mathbf{x}_{t+1} = \mathscr{M}_t(\mathbf{x}_t)$$

Flavours:

- Strong-constraint 4DVar (as above)
- Weak-constraint 4DVar (allow for model errors)
- Incremental version
- 3D-FGAT (incremental version with $\mathbf{M}_t = \mathbf{I}$)
- 3DVar ($\mathscr{M}_t(\mathbf{x}_t) = \mathbf{x}_t$).

Variational data assimilation (cont)

Properties:

- Uses observations at correct time.
- Uses dynamical model as a constraint, so can fit changes in observations over a window.
- Weak-constraint formulation allows for model error (but need to specify $\mathbf{Q}_t$).
- Assumes Gaussian prior and observations.
- $\mathbf{B}_0$ is modelled/parametrised (e.g. need control variable transforms). Not properly flow-dependent and is too simple, but can include balances.
- Analysis is sub-optimal if $\mathscr{M}$ or $\mathscr{H}$ is non-linear; can end up in a local minimum.
- Need tangent linear of $\mathscr{M}_t$ and $\mathscr{H}_t$ and their adjoints (for gradient calculation) - Difficult to develop (time and expertise). But 3D-FGAT avoids this.
- Usually does not provide information on analysis uncertainty.
- Difficult to parallelize.

The Kalman filter (and extended Kalman filter)

▷ Propagates the mean state and its error covariance sequentially; ▷ forecast/analysis is mean of the prior/posterior; ▷ the analysis is the state that has minimum variance; ▷ strong theoretical basis.

$$
\begin{aligned}
\text{forecast state: } \mathbf{x}_t^{\mathrm{f}} &= \mathscr{M}_t(\mathbf{x}_{t-1}^{\mathrm{a}}) \\
\text{forecast covariance: } \mathbf{P}_t^{\mathrm{f}} &= \mathbf{M}_t \mathbf{P}_{t-1}^{\mathrm{a}} \mathbf{M}_t^{\mathrm{T}} + \mathbf{Q}_t \\
\text{analysis state: } \mathbf{x}_t^{\mathrm{a}} &= \mathbf{x}_t^{\mathrm{f}} + \mathbf{K}_t \left(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}_t^{\mathrm{f}})\right) \\
\text{Kalman gain: } \mathbf{K}_t &= \mathbf{P}_t^{\mathrm{f}} \mathbf{H}_t^{\mathrm{T}} \left(\mathbf{H}_t \mathbf{P}_t^{\mathrm{f}} \mathbf{H}_t^{\mathrm{T}} + \mathbf{R}_t\right)^{-1} \\
\text{analysis covariances: } \mathbf{P}_t^{\mathrm{a}} &= (\mathbf{I} - \mathbf{K}_t \mathbf{H}_t)\,\mathbf{P}_t^{\mathrm{f}}
\end{aligned}
$$

The Kalman filter (and extended Kalman filter) cont.

Properties:

- Allows for correct propagation of forecast covariance matrix.
- Provides estimate of analysis error covariance.
- Allows for model error (but need to specify $\mathbf{Q}_t$).
- Assumes Gaussian prior and observations.
- Assumes $\mathscr{M}$ and $\mathscr{H}$ are linear (weak non-linearity is allowed in the extended KF).
- Unfeasible when n is large as matrices are treated explicitly.

Ensemble Kalman filters

▷ Based on KF equations; ▷ propagates N-member ensemble of forecasts to estimate $\mathbf{P}^{\mathrm{f}}_t$.

$$
\begin{aligned}
& \mathbf{x}^{(i),\mathrm{f}}_t &=& \mathscr{M}_t(\mathbf{x}^{(i),\mathrm{a}}_{t-1}) + \beta^{(i)} \\
n \times N: \quad & \mathbf{X}'^{\mathrm{f}}_t &=& \left(\mathbf{x}^{(1),\mathrm{f}}_t - \bar{\mathbf{x}}^{\mathrm{f}}_t \quad \cdots \quad \mathbf{x}^{(N),\mathrm{f}}_t - \bar{\mathbf{x}}^{\mathrm{f}}_t \right) \\
p \times N: \quad & \mathbf{Y}'_t &=& \left(\mathscr{H}_t(\mathbf{x}^{(1),\mathrm{f}}_t) - \mathscr{H}(\bar{\mathbf{x}}^{\mathrm{f}}_t) \quad \cdots \quad \mathscr{H}_t(\mathbf{x}^{(N),\mathrm{f}}_t) - \mathscr{H}_t(\bar{\mathbf{x}}^{\mathrm{f}}_t) \right) \\
n \times N: \quad & \mathbf{X}'^{\mathrm{a}}_t &=& \left(\mathbf{x}^{(1),\mathrm{a}}_t - \bar{\mathbf{x}}^{\mathrm{a}}_t \quad \cdots \quad \mathbf{x}^{(N),\mathrm{a}}_t - \bar{\mathbf{x}}^{\mathrm{a}}_t \right)
\end{aligned}
$$

Stochastic EnKF

$$
\mathbf{x}^{(i)\mathrm{a}}_t = \mathbf{x}^{(i)\mathrm{f}}_t + \mathbf{K}_t \left(\mathbf{y}_t + \varepsilon^{(i)}_y - \mathscr{H}_t(\mathbf{x}^{(i)\mathrm{f}}_t) \right)
$$

$$
\mathbf{K}_t = \mathbf{X}'^{\mathrm{f}}_t \mathbf{Y}'^{\mathrm{T}}_t \left(\mathbf{Y}'_t \mathbf{Y}'^{\mathrm{T}}_t + (N-1)\mathbf{R}_t \right)^{-1}
$$

Ensemble Transform KF

$$
\bar{\mathbf{x}}^{\mathrm{a}}_t = \bar{\mathbf{x}}^{\mathrm{f}}_t + \mathbf{K}_t \left(\mathbf{y}_t - \mathscr{H}_t(\bar{\mathbf{x}}^{\mathrm{f}}_t) \right)
$$

$$
\mathbf{X}'^{\mathrm{a}}_t = \mathbf{X}'^{\mathrm{f}}_t \mathbf{T}_t
$$

$$
\mathbf{K}_t = \mathbf{X}'^{\mathrm{f}}_t \mathbf{T}_t \mathbf{T}^{\mathrm{T}}_t \mathbf{Y}'^{\mathrm{T}}_t \mathbf{R}^{-1}_t
$$

$$
\mathbf{T}_t = \left(\mathbf{I} + \mathbf{Y}'^{\mathrm{T}}_t \mathbf{R}^{-1}_t \mathbf{Y}'_t \right)^{-1/2}
$$

Ensemble Kalman filters (cont)

Flavours

- Stochastic EnKF
- Singular Evolutive Interpolated Kalman Filter (SEIK)
- Ensemble Transform Kalman Filter (ETKF)
- Ensemble Adjustment Kalman Filter (EAKF)
- Ensemble Square Root Filter (EnSRF)
- etc.

Ensemble Kalman filters (cont)

Properties:

- $\mathscr{M}$ and $\mathscr{H}$ can be non-linear.
- Works when $N \ll n$ (but caveats).
- Avoids linear/adjoint coding.
- Easy to code.
- Parallelization is scalable with N.
- Assumes Gaussian prior and observations.
- Need localization to deal with sampling noise.
- Localization can disturb physical properties of ensemble (e.g. balance).
- Needs inflation to avoid filter divergence (ensemble under-spread).

EnVar (ensemble-variational)

▷ As variational DA, but where $\mathbf{B} \to \mathbf{X}_0'^{\mathrm{f}} \mathbf{X}_0'^{\mathrm{f}\,\mathrm{T}} / (N-1)$ from a parallel ensemble; ▷ analysis increment is a linear combination of forecast ensemble perturbations. E.g. En4DVar:

$$
\begin{aligned}
\mathbf{x}_0^{\mathrm{a}} &= \mathbf{x}_0^{\mathrm{f}} + \mathbf{X}'^{\mathrm{f}} \delta \mathbf{v}_{\mathrm{ens}} / \sqrt{N-1} \qquad \delta \mathbf{v}_{\mathrm{ens}} \text{ is an } N\text{-element vector} \\
J[\delta \mathbf{v}_{\mathrm{ens}}] &= \frac{1}{2} \delta \mathbf{v}_{\mathrm{ens}}^{\mathrm{T}} \delta \mathbf{v}_{\mathrm{ens}} + \frac{1}{2} \sum_{t=0}^{T} \left(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}_t)\right)^{\mathrm{T}} \mathbf{R}_t^{-1} (\bullet) \\
& \text{subject to } \delta \mathbf{x}_{t+1} = \mathbf{M}_t (\delta \mathbf{x}_t) \quad \text{and } \delta \mathbf{x}_0 = \mathbf{X}'^{\mathrm{f}} \delta \mathbf{v}_{\mathrm{ens}} / \sqrt{N-1} \\
& \qquad \mathbf{x}_{t+1}^{\mathrm{f}} = \mathscr{M}_t \left(\mathbf{x}_t^{\mathrm{f}}\right)
\end{aligned}
$$

EnVar (ensemble-variational) cont.

Properties:

- Has the benefits of variational DA but with a flow-dependent $\mathbf{B}$-matrix.
- Assumes Gaussian prior and observations.
- Needs localization and a separate parallel ensemble.
- En4DVar still needs the linear model and adjoint. 4DEnVar uses 4D ensembles and avoids these, but localization becomes very difficult.

Hybrid methods

As variational DA, but where

$$\mathbf{B}_0 \to (1-\beta)\mathbf{B}_0 + \beta \mathbf{X}_0'^{\mathrm{f}} \mathbf{X}_0'^{\mathrm{f}^{\mathrm{T}}} / (N-1)$$

(new matrix is full rank and flow-dependent).

Hybrid methods (cont)

Traditional 4DVar with control variable transform:

$$J[\delta \mathbf{v}_\mathrm{B}] = \frac{1}{2}\delta \mathbf{v}_\mathrm{B}^\mathrm{T}\delta \mathbf{v}_\mathrm{B} + \frac{1}{2}\sum_{t=0}^{T}\left(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}_t^\mathrm{f}) - \mathbf{H}_t\delta\mathbf{x}_t\right)^\mathrm{T}\mathbf{R}_t^{-1}(\bullet)$$

$$\text{subject to } \delta\mathbf{x}_{t+1} = \mathbf{M}_t(\delta\mathbf{x}_t), \quad \delta\mathbf{x}_0 = \mathbf{U}\delta\mathbf{v}_\mathrm{B}$$

Hybrid-En4DVar:

$$J[\delta \mathbf{v}_\mathrm{B}, \delta \mathbf{v}_\mathrm{ens}] = \frac{1}{2}\delta \mathbf{v}_\mathrm{B}^\mathrm{T}\delta \mathbf{v}_\mathrm{B} + \frac{1}{2}\delta \mathbf{v}_\mathrm{ens}^\mathrm{T}\delta \mathbf{v}_\mathrm{ens} + \frac{1}{2}\sum_{t=0}^{T}\left(\mathbf{y}_t - \mathscr{H}_t(\mathbf{x}_t^\mathrm{f}) - \mathbf{H}_t\delta\mathbf{x}_t\right)^\mathrm{T}\mathbf{R}_t^{-1}(\bullet)$$

$$\text{subject to} \quad \delta\mathbf{x}_{t+1} = \mathbf{M}_t(\delta\mathbf{x}_t), \quad \delta\mathbf{x}_0 = \sqrt{1-\beta}\,\mathbf{U}\delta\mathbf{v}_\mathrm{B} + \sqrt{\beta}\,\mathbf{X}'^\mathrm{f}\delta\mathbf{v}_\mathrm{ens}/\sqrt{N-1}$$

Hybrid methods (cont)

Properties:

- Has the benefits of variational DA but with a full-rank flow-dependent $\mathbf{B}$-matrix.
- Assumes Gaussian prior and observations.
- Still needs localization and a separate parallel ensemble.
- Can get very complex to develop.

Particle filters

▷ Non-Gaussian; ▷ approximates prior and posterior PDFs as summation of 'delta-functions'. Standard PF:

$$\begin{aligned} \text{prior PDF: } p(\mathbf{x}) &= \sum_{i=1}^{N} w_i^{\text{prior}} \delta(\mathbf{x}-\mathbf{x}_i), & \sum_{i=1}^{N} w_i^{\text{prior}} = 1/N \\ \text{posterior PDF: } p(\mathbf{x}|\mathbf{y}) &= \sum_{i=1}^{N} w_i^{\text{post}} \delta(\mathbf{x}-\mathbf{x}_i), & w_i^{\text{post}} = \frac{w_i^{\text{prior}} p(\mathbf{y}|\mathbf{x}_i)}{\sum_{i=1}^{N} w_i^{\text{prior}} p(\mathbf{y}|\mathbf{x}_i)} \end{aligned}$$

Particle filters (cont)

Properties:

- Fundamentally no need for covariance matrices - Sample from full pdf.
- No need to assume Gaussianity
- Standard PF is degenerate (weight tends to accumulate for one particle). But several approaches to try to overcome this.
- 'Resampling' still a problem for lots of obs.

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:
PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:
PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	✗	✗	✗	✗	✗	✓
Large system	✓	✗	✓	✓	✓	✓
Need info on analysis error	✗	✓	✓	✗	✗	✓
TLM/ adjoint needed	✓	✓	✗	(✓✗)	(✓✗)	✗
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	✗
Easily parallelizable	✗	✗	✓	✗	✗	✓

DA software:
PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:

PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:
PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:
PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:

PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...

Which method is right for you?

	Var	KF	EnKF	EnVar	Hybrid	PF
Non-Gaussian	X	X	X	X	X	✓
Large system	✓	X	✓	✓	✓	✓
Need info on analysis error	X	✓	✓	X	X	✓
TLM/ adjoint needed	✓	✓	X	(✓X)	(✓X)	X
Model expensive to run (no more than 50-100 runs)	✓	✓	✓	✓	✓	X
Easily parallelizable	X	X	✓	X	X	✓

DA software:

PDAF=Parallel Data Assimilation Framework;

DART=Data Assimilation Research Testbed;

DAPPER=Data assimilation package in Python for experimental research;

JEDI=Joint Effort for Data assimilation Integration;

...