EnKF-C user guide

version 0.49

Pavel Sakov

July 8, 2014

Contents

License

Chapter 1

Introduction

EnKF-C aims to provide a light-weight generic framework for off-line data assimilation (DA) into large-scale layered geophysical models with the ensemble Kalman filter (EnKF). Here “light-weight” has higher priority than “generic”; that is, the code is not designed to cover every virtual possibility for the sake of it, but rather to be expandable in practical (from the author’s point of view) situations. Following are its other main features:

- coded in C for GNU/Linux platform;
- can conduct DA either in EnKF or ensemble optimal interpolation (EnOI) mode.

To make the code simpler, EnKF-C makes two assumptions about the model:

- that it is layered;
- and that model states are stored in NetCDF format,

although the latter may be not strictly necessary.

EnKF-C is supposed to run with localisation only; at the moment there is no option for global analysis.

Chapter 2

Overview

EnKF-C conducts data assimilation in three stages: *prep*, *calc* and *post*.

prep preprocesses observations so that they are ready for DA. It has the following work flow:

- read original observations and fill each observation into a uniform structure *measurement*;
- write observations to file `observations-orig.nc`;
- combine observations into superobservations;
- write super observations to `observations.nc`.

calc calculates ensemble transforms for updating the forecast ensemble of model states (EnKF) or the background model state (EnOI) in the following stages:

- read observations from `observations.nc`;
- calculate ensemble of forecast observations $\mathbf{HE}^f$ (EnKF) or ensemble of forecast observation anomalies $\mathbf{HA}^f$ and background observation estimates $\mathbf{Hx}^f$ (EnOI);
- for each horizontal grid cell calculate local ensemble transforms $\mathbf{X}_5$ (EnKF) or background update coefficients $\mathbf{w}$ (EnOI);
- save these transforms to `X5.nc` (EnKF) or `w.nc` (EnOI);
- calculate and report forecast and analysis innovation statistics.

post updates the ensemble (EnKF) or the background (EnOI) by using transforms calculated by *calc*.

Chapter 3

EnKF

3.1 Kalman filter

The Kalman filter (KF) is the underlying concept behind the EnKF. It is a rather simple concept if formulated as the recursive least squares.

Consider a global (in time) nonlinear minimisation problem

$$\{\mathbf{x}_i^a\}_{i=1}^k = \arg \min_{\{\mathbf{x}_i\}_{i=1}^k} \mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_k), \tag{3.1}$$

$$\begin{aligned}\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_k) = \Bigg\{ & (\mathbf{x}_1 - \mathbf{x}_1^f)^{\mathrm{T}} (\mathbf{P}_1^f)^{-1} (\mathbf{x}_1 - \mathbf{x}_1^f) \\ & + \sum_{i=1}^k \left[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)\right]^{\mathrm{T}} (\mathbf{R}_i)^{-1} \left[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)\right] \\ & + \sum_{i=2}^k \left[\mathbf{x}_i - \mathcal{M}_i(\mathbf{x}_{i-1})\right]^{\mathrm{T}} (\mathbf{Q}_i)^{-1} \left[\mathbf{x}_i - \mathcal{M}_i(\mathbf{x}_{i-1})\right] \Bigg\}. \end{aligned} \tag{3.2}$$

Here $\{\mathbf{x}_i^a\}_{i=1}^k$ is a set of k state vectors that minimise the cost function (3.2); indices $i = 1, \ldots, k$ correspond to a sequence of DA cycles, so that, for example, $\mathbf{x}_1$ is the first model state and $\mathbf{x}_k$ is the last model state; $\mathbf{y}_i$ are observation vectors related to the model state by a nonlinear observation operator $\mathcal{H}_i(\mathbf{x})$; $\mathcal{M}_i(\mathbf{x})$ is a nonlinear model operator relating the model states $\mathbf{x}_{i-1}$ and $\mathbf{x}_i$; $\mathbf{P}_1^f$ is the initial state error covariance; $\mathbf{R}_i$ is the observation error covariance for $\mathbf{y}_i$; and $\mathbf{Q}_i$ is the model error covariance for $\mathcal{M}_i$; $(\cdot)^{\mathrm{T}}$ denotes matrix transposition.

The minimisation problem (3.1,3.2) is, generally, very complicated, but, luckily, has an exact solution in the *linear* case; moreover, this solution is recursive. Namely, if we assume that $\mathcal{M}$ and $\mathcal{H}$ are affine:

$$\mathcal{M}_i(\mathbf{x}^{(1)}) - \mathcal{M}_i(\mathbf{x}^{(2)}) = \mathbf{M}_i\,(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}), \tag{3.3a}$$

$$\mathcal{H}_i(\mathbf{x}^{(1)}) - \mathcal{H}_i(\mathbf{x}^{(2)}) = \mathbf{H}_i\,(\mathbf{x}^{(1)} - \mathbf{x}^{(2)}), \tag{3.3b}$$

where $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}$ are arbitrary model states, and $\mathbf{M}_i$, $\mathbf{H}_i$ = Const, then the cost function (3.2) becomes quadratic, and

$$\min_{\{\mathbf{x}_i\}_{i=1}^{k-1}} \mathcal{L}_k(\mathbf{x}_1,\dots,\mathbf{x}_k) = (\mathbf{x}_k - \mathbf{x}_k^f)^{\mathrm{T}}(\mathbf{P}_k^f)^{-1}(\mathbf{x}_k - \mathbf{x}_k^f) + \text{Const.}$$

Then

$$\min_{\{\mathbf{x}_i\}_{i=1}^{k}} \mathcal{L}_{k+1}(\mathbf{x}_1,\dots,\mathbf{x}_k,\mathbf{x}_{k+1}) = (\mathbf{x}_{k+1} - \mathbf{x}_{k+1}^f)^{\mathrm{T}}(\mathbf{P}_{k+1}^f)^{-1}(\mathbf{x}_{k+1} - \mathbf{x}_{k+1}^f) + \text{Const},$$

where

$$\mathbf{x}_{k+1}^f = \mathcal{M}_k(\mathbf{x}_k^a), \tag{3.4a}$$

$$\mathbf{P}_{k+1}^f = \mathbf{M}_k\mathbf{P}_k^a(\mathbf{M}_k)^{\mathrm{T}} + \mathbf{Q}_k, \tag{3.4b}$$

$$\mathbf{x}_k^a = \mathbf{x}_k^f + \mathbf{K}_k\left[\mathbf{y}_k - \mathcal{H}(\mathbf{x}_k^f)\right], \tag{3.5a}$$

$$\mathbf{P}_k^a = (\mathbf{I} - \mathbf{K}_k\mathbf{H}_k)\mathbf{P}_k^f, \tag{3.5b}$$

where

$$\mathbf{K}_k \equiv \mathbf{P}_k^f(\mathbf{H}_k)^{\mathrm{T}}\left[\mathbf{H}_k\mathbf{P}_k^f(\mathbf{H}_k)^{\mathrm{T}} + \mathbf{R}_k\right]^{-1}. \tag{3.5c}$$

This solution is known as the Kalman filter (KF, Kalman, 1960). Equations (3.4) describe advancing the system in time, which represents the stage commonly called “forecast”, while equations (3.5) describe assimilation of observations, which is the stage commonly called “analysis”. The superscripts f and a are used hereafter to refer to the forecast and analysis variables, correspondingly. The forecast and analysis model state estimates $\mathbf{x}^f$ and $\mathbf{x}^a$ are commonly called as (simply) forecast and analysis. Matrix $\mathbf{K}$ is called Kalman gain.

There are a few things to be noted about the KF:

1. The state $\mathbf{X}$ of the DA system (SDAS) is represented by the estimated model state vector and model state error covariance:

   $$\mathbf{X} = \{\mathbf{x}, \mathbf{P}\}. \tag{3.6}$$

   This means that in any moment of time all previous information about the system is encrypted into the current forecast or analysis SDAS.

2. The KF provides solution for the last analysis, corresponding to $\mathbf{x}_k^a$ in (3.1) (or, with a minor re-formulation, to the last forecast); finding the full (global in time) solution requires application of the Kalman *smoother* (KS). Both the KF and KS can be derived by a re-factorisation of the positive definite quadratic form (3.2).

3. Because SDAS represents a (part of a) solution of the global least squares problem, it does not depend on the order in which observations have been assimilated or on their grouping.

4. Ditto, the SDAS does not depend on a linear non-singular transform of the model state in the sense that the forward and inverse transforms commute with the evolution of the DA system.

5. Solution (3.4,3.5) can be *used* in a nonlinear case by approximating

   $$\mathbf{M}_i \leftarrow \nabla\mathcal{M}_i(\mathbf{x}_{i-1}^f),$$
   $$\mathbf{H}_i \leftarrow \nabla\mathcal{H}_i(\mathbf{x}_i^f),$$

   in which case it is called the extended Kalman filter (EKF).

3.2 EnKF

The canonic form of the KF (3.4,3.5) is not necessarily the most convenient one. The corresponding algorithms can be prone to loosing the positive definiteness of the state error covariance $\mathbf{P}$ due to round-up errors; but more importantly, explicit use of $\mathbf{P}$ makes these algorithms non-scalable in regard to the model state dimension.

Both these immediate problems can addressed with the ensemble Kalman filter, or the EnKF. In the EnKF the SDAS is carried by an ensemble of model states $\mathbf{E}$, which can be breaked into ensemble mean and ensemble anomalies:

$$\mathbf{X} = \{\mathbf{E}\} = \{\mathbf{x}, \mathbf{A}\}. \tag{3.7}$$

It is related to the SDAS of the KF (3.6) as follows:

$$\mathbf{x} = \frac{1}{m}\mathbf{E}\mathbf{1}, \tag{3.8a}$$

$$\mathbf{P} = \frac{1}{m-1}\mathbf{A}\mathbf{A}^{\mathrm{T}}, \tag{3.8b}$$

$$\mathbf{A} \equiv \mathbf{E} - \mathbf{x}\mathbf{1}^{\mathrm{T}}, \tag{3.8c}$$

where $\mathbf{1}$ is a vector with all elements equal to 1. The above means that the model state estimate is given by the ensemble mean, while the model state error covariance estimate is implicitly represented by the ensemble anomalies $\mathbf{A}$ via the factorisation (3.8b).

The EnKF is linearly scalable in regard to the state vector dimension and in some regards is a more natural and extendable representation of the KF than the canonic form (3.4,3.5). In particular, the forecast stage of the EnKF involves only propagation of each ensemble member:

$$\mathbf{E}_i^f = \mathcal{M}_i(\mathbf{E}_{i-1}^a). \tag{3.9}$$

This is a remarkably simple equation, even though the model error still needs to be handled in some way. One option is to include stochastic model error into the model above, unlike the model in the KF forecast equation (3.4a). Another option is to use multiplicative inflation. The third option – to mimic the treatment of the model error in the KF – is non-scalable in regard to the model state dimension.

At the analysis stage one has to update the ensemble mean and ensemble anomalies to match (3.5). This is different to the forecast stage, when each ensemble member is propagated individually.

Note that factorisation (3.8b) is not unique: if $\mathbf{A}$ satisfies (3.8b), then $\tilde{\mathbf{A}} = \mathbf{A}\mathbf{U}$, where $\mathbf{U}$ is an arbitrary orthonormal matrix $\mathbf{U}\mathbf{U}^{\mathrm{T}} = \mathbf{I}$, also satisfies (3.8b). However, $\tilde{\mathbf{A}}$ should not only factorise $\mathbf{P}$, but also remain an ensemble anomalies matrix, $\tilde{\mathbf{A}}\mathbf{1} = \mathbf{0}$. This requires an additional constraint $\mathbf{U}\mathbf{1} = \mathbf{1}$. Summarising, if $\mathbf{E} = \mathbf{x}\mathbf{1}^{\mathrm{T}} + \mathbf{A}$ is an ensemble that satisfies (3.8), then ensemble

$$\tilde{\mathbf{E}} = \mathbf{x}\mathbf{1}^{\mathrm{T}} + \mathbf{A}\mathbf{U}^p, \quad \mathbf{U}^p : \ \mathbf{U}^p(\mathbf{U}^p)^{\mathrm{T}} = \mathbf{I}, \ \mathbf{U}^p\mathbf{1} = \mathbf{1} \tag{3.10}$$

also satisfies (3.8); and if $\mathbf{E}$ is full rank, then (3.10) describes all possible ensembles matching a given SDAS of the KF. Such transformation of the ensemble is called ensemble *redrawing*. In the linear case (i.e. for affine model and observation operators) redrawing of the ensemble in the EnKF does not affect the evolution of the underlying KF; and conversely, in the nonlinear case the redrawing does indeed affect the evolution of the underlying KF.

3.3 EnKF analysis

In this section we will give a brief overview of solutions for the EnKF analysis, and then describe the particular schemes used in EnKF-C.

3.3.1 Overview

In the EnKF the update of the ensemble mean matches that in the KF (3.5a), although the use of ensemble anomalies instead of state error covariance can reflect in some algorithmic differences.

The update of ensemble anomalies should implicitly match the update of covariance in the KF (3.5b). Technically, this can be done either via a right-multiplied or left-multiplied transforms of the ensemble anomalies:

$$\mathbf{A}^a = \mathbf{T}_L\, \mathbf{A}^f, \tag{3.11}$$

or

$$\mathbf{A}^a = \mathbf{A}^f\, \mathbf{T}_R. \tag{3.12}$$

$\mathbf{T}_L$ and $\mathbf{T}_R$ are referred to hereafter as left-multiplied and right-multiplied ensemble transform matrices (ETMs), respectively. Note that for a full rank ensemble $\mathbf{T}_R$ has to satisfy $\mathbf{T}_R\mathbf{1} = \mathbf{1}$. It follows from (3.10) that if $\mathbf{T}_R$ is a particular solution for the right multiplied ETM, then (for a full rank ensemble) any other solution can be written as

$$\tilde{\mathbf{T}}_R = \mathbf{T}_R\mathbf{U}^p, \quad \mathbf{U}^p:\ \mathbf{U}^p(\mathbf{U}^p)^{\mathrm{T}} = \mathbf{I},\ \mathbf{U}^p\mathbf{1} = \mathbf{1}. \tag{3.13}$$

Similarly, the analysis increment can be represented as a linear combination of the ensemble anomalies:

$$\mathbf{x}^a = \mathbf{x}^f + \mathbf{A}^f\mathbf{w}. \tag{3.14}$$

Equations (3.12) and (3.14) can be combined into a single transform of the ensemble:

$$\mathbf{E}^a = \mathbf{E}^f\mathbf{X}_5, \tag{3.15}$$

$$\mathbf{X}_5 = \frac{1}{m}\mathbf{1}\mathbf{1}^{\mathrm{T}} + \left(\mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\left(\mathbf{w}\mathbf{1}^{\mathrm{T}} + \mathbf{T}_R\right) = \frac{1}{m}\mathbf{1}\mathbf{1}^{\mathrm{T}} + \mathbf{w}\mathbf{1}^{\mathrm{T}} + \left(\mathbf{I} - \frac{1}{m}\mathbf{1}\mathbf{1}^{\mathrm{T}}\right)\mathbf{T}_R, \tag{3.16}$$

as $\mathbf{1}^{\mathrm{T}}\mathbf{w} = 0$. For many common schemes, including the ETKF and DEnKF, $\mathbf{1}^{\mathrm{T}}\mathbf{T}_R = \mathbf{1}^{\mathrm{T}}$, so that

$$\mathbf{X}_5 = \mathbf{w}\mathbf{1}^{\mathrm{T}} + \mathbf{T}_R. \tag{3.17}$$

The designation $\mathbf{X}_5$ is used for historic reasons, following Evensen (2003).

3.3.2 Some schemes

As follows from the previous section, there are multiple solutions for the ETM that match the KF covariance update equation (3.5b); however the particular solutions may have different properties in practice due to the DAS nonlinearity, their algorithmic convenience, or their robustness in suboptimal conditions. This section provides some background for the schemes used in EnKF-C:

- ETKF;
- DEnKF.

ETKF

It is easy to show using the definition of $\mathbf{K}$ (3.5c) and matrix shift lemma that

$$(\mathbf{I}-\mathbf{KH})\,\mathbf{P}^f = (\mathbf{I}-\mathbf{KH})^{1/2}\,\mathbf{P}^f\,(\mathbf{I}-\mathbf{KH})^{\mathrm{T}/2},$$

which yields the following solution for the left-multiplied ETM:

$$\mathbf{T}_L = (\mathbf{I}-\mathbf{KH})^{1/2} \tag{3.18}$$

(Sakov and Oke, 2008b), or

$$\mathbf{A}^a = (\mathbf{I}-\mathbf{KH})^{1/2}\mathbf{A}^f.$$

Hereafter by $\mathbf{X}^{1/2}$ we denote the unique positive definite square root from a positive definite (generally, non-symmetric) matrix $\mathbf{X}$, defined as $\mathbf{X}^{1/2} = \mathbf{V}\mathbf{L}^{1/2}\mathbf{V}^{-1}$, where $\mathbf{X} = \mathbf{V}\mathbf{L}\mathbf{V}^{-1}$ is the eigenvalue decomposition of $\mathbf{X}$. By "matrix shift lemma" we refer to the following identity:

$$\mathcal{F}(\mathbf{AB})\,\mathbf{A} = \mathbf{A}\,\mathcal{F}(\mathbf{BA}), \tag{3.19}$$

where $\mathcal{F}$ is an arbitrary function expandable into Taylor series. Rewriting this as

$$\mathbf{A}^a = \left[\mathbf{I} - \frac{1}{m-1}\mathbf{A}^f(\mathbf{HA}^f)^{\mathrm{T}}(\mathbf{HP}^f\mathbf{H}^{\mathrm{T}}+\mathbf{R})^{-1}\mathbf{H}\right]^{1/2}\mathbf{A}^f$$

and using the matrix shift lemma, we obtain:

$$\mathbf{A}^a = \mathbf{A}^f\left[\mathbf{I} - \frac{1}{m-1}(\mathbf{HA}^f)^{\mathrm{T}}(\mathbf{HP}^f\mathbf{H}^{\mathrm{T}}+\mathbf{R})^{-1}\mathbf{HA}^f\right]^{1/2}$$

which yields the corresponding to (3.18) right multiplied ETM:

$$\mathbf{T}_R = \left[\mathbf{I} - \frac{1}{m-1}(\mathbf{HA}^f)^{\mathrm{T}}(\mathbf{HP}^f\mathbf{H}^{\mathrm{T}}+\mathbf{R})^{-1}\mathbf{HA}^f\right]^{1/2} \tag{3.20}$$

(Evensen, 2004). Applying the matrix inversion lemma

$$(\mathbf{A}+\mathbf{ULV})^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{U}(\mathbf{L}^{-1}+\mathbf{VA}^{-1}\mathbf{U})^{-1}\mathbf{VA}^{-1},$$

(3.18) can be transformed to:

$$\mathbf{T}_L = (\mathbf{I} + \mathbf{P}^f \mathbf{H}^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{H})^{-1/2} \tag{3.21}$$

(Sakov and Bertino, 2011); and applying the matrix shift lemma yields the corresponding right multiplied ETM:

$$\mathbf{T}_R = \left[\mathbf{I} + \frac{1}{m-1}(\mathbf{HA})^{\mathrm{T}} \mathbf{R}^{-1} \mathbf{HA}\right]^{-1/2}, \tag{3.22}$$

also known as the ensemble transform Kalman filter, or ETKF (Bishop et al., 2001).

Solution (3.22) can be described as the (unique) symmetric right multiplied solution. It represents the minimum distance solution in the sense that the corresponding analysis ensemble is the closest one to the forecast ensemble, with the inverse forecast (or analysis) covariance as a norm (Ott et al., 2003, rev. 2005). In this sense the ETKF solution and equivalent solutions (3.18–3.21) provide the best continuity between the forecast and analysis ensembles.

Note that while the left multiplied solutions (3.18,3.21) correspond to symmetric right multiplied solution, they are not symmetric.

In a typical DAS with a large scale model one can expect $m = 100$, $p = 10^3 - 10^7$, $n = 10^6 - 10^9$; that is

$$m \ll p \ll n. \tag{3.23}$$

Therefore, considering the size of ETMs ($n \times n$ for the left-multiplied ETMs and $m \times m$ for the right multiplied ETMs), only right multiplied solutions are suitable for use with large scale models. The ETKF solution (3.22) represents the most popular option due to its simple form and numerical effectiveness: for a diagonal $\mathbf{R}$, it only requires to calculate inverse square root of an $m \times m$ matrix.

DEnKF

Assuming that $\mathbf{KH}$ is small in some sense, one can approximate solution (3.18) by expanding it into Taylor series about $\mathbf{I}$ and keeping the first two terms of the expansion:

$$\mathbf{T}_L = \mathbf{I} - \frac{1}{2}\mathbf{KH}. \tag{3.24}$$

This approximation is known as the deterministic Kalman filter, or DEnKF (Sakov and Oke, 2008a). It has a simple interpretation of updating the ensemble anomalies using half of the Kalman gain, but apart from that the DEnKF often represents a good practical choice due to its algorithmic convenience and good performance in suboptimal situations. DEnKF is the default scheme in EnKF-C.

3.3.3 Some numerical considerations

Instead of using the forecast ensemble observation anomalies $\mathbf{HA}^f$ and innovation $\mathbf{y} - \mathcal{H}(\mathbf{x}^f)$ it is convenient to use their standardised versions:

$$\mathbf{s} = \mathbf{R}^{-1/2} \left[\mathbf{y} - \mathcal{H}(\mathbf{x}^f)\right] / \sqrt{m-1}, \tag{3.25}$$

$$\mathbf{S} = \mathbf{R}^{-1/2}\mathbf{HA}^f / \sqrt{m-1}. \tag{3.26}$$

Then

$$\mathbf{w} = \mathbf{Gs}, \tag{3.27}$$

for the ETKF

$$\mathbf{T}_R = (\mathbf{I} + \mathbf{S}^{\mathrm{T}}\mathbf{S})^{-1/2}, \tag{3.28}$$

and for the DEnKF

$$\mathbf{T}_R = \mathbf{I} - \frac{1}{2}\mathbf{GS}, \tag{3.29}$$

where

$$\mathbf{G} \equiv (\mathbf{I} + \mathbf{S}^{\mathrm{T}}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}}, \tag{3.30}$$

$$= \mathbf{S}^{\mathrm{T}}(\mathbf{I} + \mathbf{SS}^{\mathrm{T}})^{-1}. \tag{3.31}$$

Here (3.30) involves inversion of a $m \times m$ matrix, while (3.31) involves inversion of a $p \times p$ matrix. Therefore, in the DEnKF it is possible to calculate $\mathbf{w}$ and $\mathbf{T}$ using a single inversion of either a $p \times p$ or $m \times m$ matrix, depending on the relation between the number of observations and the ensemble size. In contrast, the ETKF (3.28) involves calculation of the inverse square root of a $m \times m$ matrix. Therefore, one can use expression (3.30) for $\mathbf{G}$ and calculate both inversion in it and inverse square root in (3.28) from the same singular value decomposition (SVD). This makes the DEnKF somewhat more numerically effective because, firstly, one can exploit situations when $p < m$ to invert a matrix of lower dimension and, secondly, it requires only matrix inversion, which can be done via a more effective Cholesky decomposition instead of SVD.

3.3.4 Localisation

Localisation is a necessary attribute of the EnKF systems with large-scale models, aimed at overcoming the rank deficiency of the ensemble. It can also be seen as aimed at removing spurious long range correlations occurring with ensembles of insufficient size.

There are two common localisation methods for the EnKF – covariance localisation (CL, Hamill and Whitaker, 2001; Houtekamer and Mitchell, 2001) and local analysis (LA, Evensen, 2003; Ott et al., 2003, rev. 2005). Although CL may have advantages in certain situations (non-local observations, "strong" assimilation), in practice the two methods produce similar results (Sakov and Bertino, 2011). For algorithmic reasons EnKF-C uses LA.

Instead of calculating a global ensemble transform $\mathbf{X}_5$, LA involves calculating local ensemble transforms $\overset{i}{\mathbf{X}}_5$ for each element i of the state vector. This is done using local normalised ensemble observation anomalies $\overset{i}{\mathbf{S}}$ and local normalised innovation $\overset{i}{\mathbf{s}}$, obtained by tapering global $\mathbf{S}$ and $\mathbf{s}$:

$$\overset{i}{\mathbf{s}} \equiv \mathbf{s} \circ \overset{i}{\mathbf{f}}, \tag{3.32a}$$

$$\overset{i}{\mathbf{S}} \equiv \mathbf{S} \circ (\overset{i}{\mathbf{f}}\, \mathbf{1}^{\mathrm{T}}), \tag{3.32b}$$

where $\mathbf{A} \circ \mathbf{B}$ denotes by-element, or Hadamard, or Schur product of matrices $\mathbf{A}$ and $\mathbf{B}$. We consider non-adaptive localisation only, when the vector of taper coefficients $\mathbf{f}$ does not depend on the SDAS or observations. Typically, the taper coefficient for observation o is a function of locations of the state element i (denoted as $\overset{i}{\mathbf{r}}$) and observation o (denoted as $\overset{\{o\}}{\mathbf{r}}$): $\overset{i}{\mathbf{f}}_o = g(\overset{i}{\mathbf{r}}, \overset{\{o\}}{\mathbf{r}})$, where g is the taper function. In layered geophysical models g is often assumed to depend only on horizontal distance between these locations:

$$\overset{i}{\mathbf{f}}_o = g(|\overset{i}{\boldsymbol{\rho}} - \overset{\{o\}}{\boldsymbol{\rho}}|), \tag{3.33}$$

or on combination of horizontal and vertical distances: $\overset{i}{f}_o = g_{xy}(|\overset{i}{\boldsymbol{\rho}} - \overset{\{o\}}{\boldsymbol{\rho}}|) g_z(|\overset{i}{z} - \overset{\{o\}}{z}|)$, where $\mathbf{r} = (\boldsymbol{\rho}, z)$, and $\boldsymbol{\rho} = (x, y)$. In the case (3.33) for a given set of observations the ensemble transform $\overset{i}{\mathbf{X}}_5$ depends only on horizontal coordinates of the state element $\mathbf{x}_i$ and can be used for updating all state elements with the same horizontal coordinates. It is currently the only option in EnKF-C.

Smooth taper functions have advantage over non-smooth functions (such as the boxcar, or step function) because they maintain the spatial continuity of the analysis. EnKF-C uses the popular polynomial taper function by Gaspari and Cohn (1999), which has a number of nice properties.

EnKF-C is designed to use horizontal localisation only. In this case a taper coefficient is a function of horizontal distance between the observation and state vector element. A local ensemble transform $\overset{i}{\mathbf{X}}_5$ calculated for some node of the horizontal grid can be applied for updating elements of every layer of the model state with the same horizontal coordinates.

3.4 Asynchronous DA

Observations assimilated at each cycle in the KF are assumed to be made simultaneously at the time of assimilation. Such observations are referred to as *synchronous*. In reality, the observations assimilated at a given cycle are made over some period of time called data assimilation window (DAW). If the DA method accounts for the time of observations, the observations are referred to as *asynchronous*. Instead of speaking about synchronous or asynchronous observations it is also common to speak about synchronous or asynchronous DA.

The EnKF can be naturally extended for the asynchronous DA. Let us consider the linear case (3.3) with perfect model ($\mathbf{Q} = 0$). The solution of the minimisation problem (3.1) is then effectively given by the initial state $\mathbf{x}_1$, while the rest of the global solution can be obtained by propagating initial state: $\mathbf{x}_2 = \mathcal{M}_2(\mathbf{x}_1),\ \mathbf{x}_3 = \mathcal{M}_3 \circ \mathcal{M}_2(\mathbf{x}_1), \dots$. The cost function

$$\mathcal{L}(\mathbf{x}_1, \dots, \mathbf{x}_k) = (\mathbf{x}_1 - \mathbf{x}_1^f)^{\mathrm{T}} (\mathbf{P}_1^f)^{-1} (\mathbf{x}_1 - \mathbf{x}_1^f) + \sum_{i=1}^{k} \left[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)\right]^{\mathrm{T}} (\mathbf{R}_i)^{-1} \left[\mathbf{y}_i - \mathcal{H}_i(\mathbf{x}_i)\right]$$

can then be written as:

$$\mathcal{L}(\mathbf{x}_1, \ldots, \mathbf{x}_k) = (\mathbf{x} - \mathbf{x}^f)^{\mathrm{T}} (\mathbf{P}^f)^{-1} (\mathbf{x} - \mathbf{x}^f) + [\mathbf{y} - \mathcal{H} \circ \mathcal{M}(\mathbf{x})]^{\mathrm{T}} \, \mathbf{R}^{-1} \, [\mathbf{y} - \mathcal{H} \circ \mathcal{M}(\mathbf{x})] \,, \qquad (3.34)$$

where we have dropped the index 1, observations $\mathbf{y}$ represent the augmented observation vector: $\mathbf{y} = [\mathbf{y}_1^{\mathrm{T}}, \ldots, \mathbf{y}_k^{\mathrm{T}}]^{\mathrm{T}}$, $\mathbf{R}$ is the corresponding observation error covariance, and operator $\mathcal{H} \circ \mathcal{M}$ is assumed to project the initial state $\mathbf{x}$ to observation space.

The cost function (3.34) has the same form as that for a single DA cycle. Consequently, one can use the EnKF solutions for $\mathbf{w}$ and $\mathbf{T}$ from section 3.3.3 calculated by using augmented normalised innovation and forecast ensemble anomalies:

$$\mathbf{s} = \mathbf{R}^{-1/2} \left[\mathbf{y} - \mathcal{H} \circ \mathcal{M}(\mathbf{x}^f) \right] / \sqrt{m-1}, \qquad (3.35)$$

$$\mathbf{S} = \mathbf{R}^{-1/2} \, \mathbf{H} \circ \mathbf{M} \, \mathbf{A}^f / \sqrt{m-1}. \qquad (3.36)$$

This means that to account for the time of observations in the EnKF one needs simply to use the ensemble at the time of the observation when calculating the corresponding elements of innovation and forecast ensemble observation anomalies. There are no specific restrictions on $\mathbf{R}$, so that in theory observation errors can be correlated in time.

Minimising (3.34) yields the updated state estimate at the beginning of the cycle. To obtain the updated state estimate at other time (e.g. at the end of the cycle) requires propagating this solution to that time. The same applies to the state error covariance: obtaining the SDAS at any particular time requires propagating the updated ensemble from the beginning of the cycle to that time. In reality, because for a linear model the operations of ensemble propagation and linear transform in ensemble space do commute, $\mathcal{M}(\mathbf{E}) \, \mathbf{X}_5 = \mathcal{M}(\mathbf{E} \, \mathbf{X}_5)$ and $(\mathbf{MA})\mathbf{T} = \mathbf{M}(\mathbf{AT})$, the ensemble transform $\mathbf{X}_5$ can be applied to the ensemble at *any* particular time (Evensen and van Leeuwen, 2000). This eliminates the need for the additional propagation.

3.5 EnOI

The EnOI, or ensemble optimal interpolation (Evensen, 2003), can be defined as the EnKF with a static or, more generally, pre-defined, ensemble. It can be summarised as follows:

$$\mathbf{x}_i^b = \mathcal{M}_i(\mathbf{x}_{i-1}^a), \qquad (3.37)$$

$$\mathbf{x}_i^a = \mathbf{x}_i^f + \mathbf{A}^b \mathbf{w}_i, \qquad (3.38)$$

where $\mathbf{x}^b$ is the forecast model state estimate referred to as the background, and $\mathbf{A}^b$ is an ensemble of static, or background, anomalies; the corresponding state error covariance $\mathbf{P}^b$ is also often referred to as the background covariance.

The main incentive for using the EnOI is its low computational cost due to the integration of only one instance of the model. Despite of the similarity with the EnKF, the EnOI is a rather different concept, as there is no global in time cost function associated with it. Conceptually the EnOI is closer to 3D-Var, as both methods use static (anisotropic, multivariate) covariance. It is an improvement on the optimal interpolation, which typically uses isotropic, homogeneous and univariate covariance.

Compared to the EnKF, due to the use of a static ensemble the EnOI avoids problems related to the ensemble spread; but at the same time it does critically depend on the ensemble, while the EnKF with a stochastic model typically "forgets" the initial ensemble over time.

Chapter 4

Technical description

4.1 Starting up: the example

It may be a good idea to start getting familiar with the system by running the example. Due to its size (over 600 MB) it is available for download separately from the EnKF-C code.

The example has been put up based on the runs of regional EnKF and EnOI reanalysis systems for Tasman Sea developed by Bureau of Meteorology. It allows one to conduct a single assimilation for 23 December 2007 (day 6565 since 1 January 1990) with either EnKF or EnOI.

To reduce the size of the system, the model state has been stripped down to two vertical levels.

To run the example, one needs to compile EnKF-C and copy executables from `bin` directory to the upper level directory of the example. To assimilate with the EnKF, one needs to run "make enkf"; to assimilate with the EnOI - "make enoi"; and to clean up after either of the above - "make clean".

The results of the analysis with the EnKF will be of form `ensemble_6565/mem<num>_<var>.nc.analysis`, where `<num>` is the member id, between 001 and 096, and `<var>` is one of the state variables: eta_t, temp, salt, u or v; the results with the EnOI will be of form `background/bg_<var>.nc.analysis`.

4.2 Parameter files

EnKF-C requires 4 parameter files to run: the main parameter file and referred by it model, grid and observation parameter files. Examples of these parameter files can be found in `examples/1`.

4.2.1 Model parameter file

Currently the model parameter file is not doing much, and exists mainly for the future expandability of the code. It defines the name of the model and the model type. The model type is a tag that specifies the custom model code in EnKF-C. The model parameter file may also contain additional parameters required by the model of a particular type, such as the mean sea level for assimilation of sea level anomalies.

4.2.2 Grid parameter file

The grid parameter file contains the grid name, grid data file and names of the dimensions and coordinates in the grid data file. It also contains the model related information, such as depth and number of layers in a vertical column for ocean models running on z grids.

The split between the parameter files is not final and may change in future.

4.2.3 Observation parameter file

Observation parameter file specifies the input observations to be assimilated. It contains an arbitrary number of independent entries, each specifying the observation type, input files, reader and, possibly, observation error, e.g.:

```
product == CARS
type = TEM
reader = standard
file=/short/p93/pxs599/obs/TS-SEP-2011/y2006/m08/cars_temp_d26.nc
file=/short/p93/pxs599/obs/TS-SEP-2011/y2006/m08/cars_temp_d27.nc
file=/short/p93/pxs599/obs/TS-SEP-2011/y2006/m08/cars_temp_d28.nc
file=/short/p93/pxs599/obs/TS-SEP-2011/y2006/m08/cars_temp_d29.nc
file=/short/p93/pxs599/obs/TS-SEP-2011/y2006/m08/cars_temp_d30.nc
error_std = 0.5
```

Observation files can be defined using wildcards "*" and "?". The available readers are listed by the variable `allreaders` defined in `prep/allreaders.c`.

The observation time only matters if the observation type is specified to be "asynchronous" (see sec. 4.4.2). In this case the model estimation for the observation is made by using model state at the appropriate time. Otherwise, observations are assumed to be made at the time of assimilation, regardless of the actual observation time.

4.3 File name conventions

EnKF-C assumes that the ensemble and background file names have some predefined formats. The file name for member `mid` and model variable `varname` assumed to be `sprintf("mem%03d_%s.nc", mid, varname)`. The background file for variable `varname` is assumed to be `sprintf("bg_%s.nc", varname)`. The above names are used for reading forecast states for synchronous DA and for writing analyses, in the case if the analyses are appended to forecasts (true by default). For asynchronous DA the member and background file names for the time slot `t` are assumed to be `sprintf("mem%03d_%s_%d.nc", mid, varname, t)` and `sprintf("bg_%s_%d.nc", varname, t)`, correspondingly.

4.4 *prep*

prep is the first stage of data assimilation in EnKF-C. Its role is to preprocess the observations by bringing them to a common form and superobing (merging close observations).

By design, *prep* is supposed to be light-weight, so that it does not read either the ensemble or background, and the only model information it needs is the model grid. (Note that this may require some additional processing at later stages for models with dynamic grid, such as HYCOM.)

The name of the binary (executable) for *prep* is `enkf_prep`. It has the following usage and options:

```
>./bin/enkf_prep
  Usage: enkf_prep <prm file> [<options>]
  Options:
  --log-all-obs
      put all obs into observations-orig.nc (default: local obs only)
  --describe-prm-format
      describe format of the parameter file and exit
  --describe-superob <sob #>
      print composition of this superobservation and exit
  --version
      print version and exit
```

`enkf_prep` writes the preprocessed observations into file `observatons.nc`. It also writes file `observatons-orig.nc`, which contains original (not superobed) observations. By default, only observations within the model grid are written to it, but the the option `--log-all-obs` changes this behaviour to writing all observations from the input files.

4.4.1 Observation types, providers, instruments

Each observation has a number of attributes defined by the fields of the structure `measurement`. One of them is observation type, which is a tag characterising the observation in a general way. For example, typical oceanographic observations may have tags SLA (for sea level anomalies), SST (sea surface temperature), TEM (subsurface temperature) and SAL (subsurface salinity). Each observation type is related to a certain model variable by the entry `OBS2VAR` in the parameter file, e.g.:

```
OBS2VAR = SST temp SLA eta_t TEM temp SAL salt
```

Different types can be related to the same model variable, as do SST and TEM in the above example.

An observation is also characterised by a provider. A “provider” can be a tag for an organisation that provides data from certain observational platforms.

The observational data from a provider can be collected by a number of instruments. The corresponding field in the `measurement` structure is supposed to be filled by the observation reader.

4.4.2 Asynchronous DA

An observation type can be specified as "asynchronous" by the `ASYNCHRONOUS` entry in the parameter file, e.g.:

```
ASYNCHRONOUS = SLA 1 SST 1
```

The above means that SLA and SST observations are considered to be asynchronous, with the time quantification of 1 day. If, for example the assimilation time is specified as "6085.5 days since 1990-01-01", then the SLA and SST observations will be binned into 1-day time intervals centred at the time of assimilation, i.e. from day 6080.0 to day 6081.0, 6081.0 to 6082.0, and so on, and estimated versus the corresponding model states. If the assimilation time were specified as "6085.5 days since 1990-01-01", then the bins would be from day 6079.5 to day 6080.5 and so on.

The model states used to calculate forecast observations are matched by file names, which are supposed to be of the form `mem<xxx>_<variable name>_<time shift>.nc` (for the EnKF) or `bg_<variable name>_<time shift>.nc` (for the EnOI). Here "time shift" is the number of the bin, with "0" corresponding to the bin centred at the time of assimilation, "-1" to the previous bin, "1" to the next bin, and so on. If the corresponding members (or the background files, in the case of EnOI) are found, the observations are assimilated asynchronously; if they are not found, then the observations are assimilated synchronously. This can be tracked from the *calc* log file, e.g.:

```
  calculating ensemble observations:
  2014-03-22 06:28:28
    ensemble size = 96
    distributing iterations:
      all processes get 6 iterations
      process 0: 0 - 5
    SST |aaaaaa|aaaaaa|aaaaaa|aaaaaa|aaaaaa
    SLA |aaaaaa|aaaaaa|aaaaaa|aaaaaa|aaaaaa
    TEM ......
    SAL ......
```

The entries "a" mean that the observations are assimilated asynchronously. They would be replaced by "s" if assimilated synchronously. The vertical lines indicate the time slots for asynchronous DA; in the above example the DAW has 5 time slots. The entries "." indicate calculating ensemble observations for synchronous observations.

4.4.3 Superobing

"Superobing" is the process of reduction of the number of observations by merging close observations before their assimilation. EnKF-C merges observations if:

- they belong to the same model grid cell;
- are of the same type;
- for asynchronous observations – belong to the same time slot.

The horizontal size of superobing cells can be increased from the default of 1 model grid cell to $N \times N$ cells by setting `SOBSTRIDE = <N>` in the parameter file; the vertical size is always equal to 1 layer.

The observations are merged by averaging their values, coordinates and times with weights inversely proportional to the observation error variance. The observation error variance of a superobservation is set to the inverse of the sum of inverse observation error variances of the merged observations. The provider and instrument fields of the superobservation are set either to those of the merged observations or to -1, depending on whether the merged observations have the same values for these fields or not.

4.5 *calc*

calc is the second stage of data assimilation in EnKF-C. It calculates 2D arrays of local ensemble transforms $\mathbf{X}_5$ or coefficients $\mathbf{w}$.

The name of the binary for *calc* is `enkf_calc`. It has the following usage and options:

```
>./bin/enkf_calc
  Usage: enkf_calc <prm file> [<options>]
  Options:
  --describe-prm-format
      describe format of the parameter file and exit
  --forecast-stats-only
      calculate and print forecast observation stats only
  --no-mean-update
      update ensemble anomalies only
  --single-observation-xyz <lon> <lat> <depth> <type> <value> <std>
      assimilate single observation with these parameters
  --single-observation-ijk <fi> <fj> <fk> <type> <value> <std>
      assimilate single observation with these parameters
  --use-these-obs <obs file>
      assimilate observations from this file; the file format must be compatible
      with that of observations.nc produced by 'enkf_prep'
  --version
      print version and exit
```

The option `--forecast-stats-only` can be used for quick calculation of the innovation statistics for a given background (or ensemble). This can be used, for example, for obtaining the persistence statistics, that is, the innovation statistics for the previous analysis.

The options `--single-observation-xyz` and `--single-observation-ijk` provide an easy way to conduct the so called single observation experiments, with the observation coordinates provided either in spatial or grid coordinates, correspondingly. The `<value>` parameter defines innovation rather than the observation value.

4.5.1 Interpolation of ensemble transforms

Local ensemble transforms $\mathbf{X}_5$ (EnKF) have or local ensemble weights $\mathbf{w}$ (EnOI) represent a smooth field with the characteristic spatial variability scale of the localisation radius. This allows to reduce computational load in *calc* by calculating local transforms or weights on a subgrid with a specified stride only and using linearly interpolated transforms or weights in the intermediate grid cells. The value of the stride is defined by the `STRIDE` entry in the parameter file.

4.5.2 Observation functions

Model estimations for observations of each type are calculated using observation functions specified for this type by entry `HFUNCTIONS` in the parameter file, e.g.:

```
HFUNCTIONS = SST standard SLA standard TEM standard SAL standard
```

The available functions for each observation type are specified by the variable `allhentries` in `calc/allhs.c`. The "standard" functions do normally perform 2D or 3D linear interpolation from the corner model grid nodes for the cell containing the observation.

4.5.3 Innovation statistics

In its course *calc* calculates some basic innovation statistics: number of observations, mean absolute forecast innovation, mean absolute analysis innovation, mean forecast innovation, mean analysis innovation, mean forecast ensemble spread, and mean analysis ensemble spread. This statistics is provided for each region defined in the parameter file, as well as for each time slot defined for asynchronous DA, and for each instrument. In addition, for 3D observations *calc* also calculates observation statistics for observations shallower or deeper than depth (height) specified by `DEPTH_SHALLOW` and `DEPTH_DEEP` in `calc/obsstats.c`. For example:

```
 printing observation statistics:
   region obs.type   # obs.  |for.inn.| |an.inn.|   for.inn.   an.inn.  for.spread  an.spread
   ------------------------------------------------------------------------------------------
   Tasman
          SST         25928    0.363       0.244     -0.077     -0.048       0.352      0.251
           -4          7214    0.303       0.187     -0.077     -0.022       0.297      0.212
           -3          4727    0.339       0.205     -0.037      0.006       0.389      0.263
           -2          4862    0.370       0.283     -0.105     -0.073       0.378      0.268
           -1          4606    0.426       0.306     -0.160     -0.143       0.377      0.275
            0          4519    0.413       0.268     -0.007     -0.023       0.348      0.257
            AVHRR     10776    0.345       0.174     -0.053     -0.031       0.336      0.227
            WindSat   14332    0.381       0.301     -0.100     -0.064       0.362      0.269
            N/A         820    0.293       0.160     -0.009      0.009       0.377      0.247
          SLA          4637    0.063       0.035      0.014      0.003       0.047      0.029
           -4           728    0.049       0.032      0.006     -0.003       0.039      0.025
           -3          1417    0.055       0.032      0.008      0.004       0.050      0.031
           -2           520    0.063       0.038      0.047      0.025       0.035      0.023
           -1           612    0.074       0.043      0.013     -0.001       0.040      0.025
            0          1360    0.073       0.036      0.012     -0.000       0.056      0.034
```

g1	1652	0.058	0.039	-0.031	-0.024	0.044	0.029
j1	1581	0.061	0.028	0.035	0.014	0.043	0.026
n1	1357	0.071	0.040	0.045	0.025	0.054	0.034
N/A	47	0.060	0.021	-0.001	0.001	0.050	0.030
TEM	284	0.421	0.253	-0.119	-0.023	0.410	0.266
ARGO	284	0.421	0.253	-0.119	-0.023	0.410	0.266
0-50m	58	0.409	0.201	-0.013	0.105	0.315	0.215
>500m	70	0.272	0.214	0.093	0.064	0.293	0.201

This excerpt shows innovation statistics for the region "Tasman". It contains sections for SST, SLA and TEM observations. The summary statistics for each observation type is shown at the top of each section; then statistics for days -4, -3, -2, -1 and 0 of a 5-day DAW are shown for the two asynchronous types, SST and SLA. After that, statistics for particular instruments is shown; "N/A" corresponds to superobservations resulted from merging observations from two or more instruments. For subsurface temperature also statistics for shallow (0–50 m) and deep (0–500 m) observations is given.

The analysis innovation statistics is calculated from the updated (analysis) ensemble observations, thus avoiding the need to access analysis files produced later by *post*. The update of ensemble observations is performed in the same way as that of any other element of the state vector: for the EnKF – by applying the appropriate local ensemble transforms to the forecast ensemble observations,

$$\mathcal{H}(\mathbf{E}^a) \leftarrow \mathcal{H}(\mathbf{E}^f)\,\mathbf{X}_5;$$

and for the EnOI – by applying the appropriate local linear combination of the ensemble observation anomalies:

$$\mathcal{H}(\mathbf{E}^a) \leftarrow \mathcal{H}(\mathbf{x}^f)\,\mathbf{1}^{\mathrm{T}} + \mathbf{H}\mathbf{A}^f\mathbf{w}.$$

4.5.4 Impact of observations

In the course of its work *calc* routinely calculates two metrics for assessing the impact of observations, degrees of freedom of signal (DFS) and spread reduction factor (SRF):

$$\mathrm{DFS} = \mathrm{tr}(\mathbf{KH}) = \mathrm{tr}(\mathbf{GS}),$$
$$\mathrm{SRF} = \sqrt{\frac{\mathrm{tr}(\mathbf{HP}^f\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1})}{\mathrm{tr}(\mathbf{HP}^a\mathbf{H}^{\mathrm{T}}\mathbf{R}^{-1})}} - 1 = \sqrt{\frac{\mathrm{tr}(\mathbf{S}^{\mathrm{T}}\mathbf{S})}{\mathrm{tr}(\mathbf{GS})}} - 1,$$

where $\mathrm{tr}(\cdot)$ is the trace function. The values of these metrics for each local analysis, calculated both for all observations and for observations of each type only, are written to file `enkf_diagn.nc`. Note that the in EnKF-C DFS and SRF are calculated from the above expressions and represent theoretical values for the EnKF analysis; they coincide with the actual DFS and SRF values only for the ETKF, but not for the DEnKF, which is an approximation of the KF (and indeed not for the EnOI, which is not even an approximation).

In the EnKF context DFS is a useful indicator of potential rank problems. Normally, it should not exceed a fraction (a half, or better, a quarter) of the ensemble size. SRF shows the "strength" of DA. "Strong" DA implies a close to optimal system, which indeed never happens in practice. Therefore, ideally, SRF should be small (below 1, on average).

4.6 *post*

post is the third and final stage of data assimilation in EnKF-C. It updates the ensemble (EnKF) or the background (EnOI) by applying the transforms calculated by *calc*.

The name of the binary for *post* is `enkf_post`. It has the following usage and options:

```
>./bin/enkf_post
  Usage: enkf_post <prm file> [<options>]
  Options:
  --calculate-spread
      calculate ensemble spread and write to spread.nc
  --describe-prm-format
      describe format of the parameter file and exit
  --direct-write
      write fields directly to the output file (default: write to tiles first)
  --leave-tiles
      do not delete tiles
  --output-increment
      output analysis increment (default: output analysis)
  --separate-output
      write results to new files (default: append to forecast files)
  --version
      print version and exit
```

The option `--separate-output` tells *post* to write the updated ensemble (EnKF) or background (EnOI) to separate analysis files, rather than appending them to the forecast files. When writing to a separate file, the same variable names are used for the analysis as for the forecast; the new files have an extra suffix `.analysis` or `.increment`, depending on whether the analysis or increment is written. By default, *post* writes the results to the forecast files by creating new variable names from the old names with an additional suffix `_an`.

By default, *post* first writes each updated horizontal fields of the model to a separate file (referred to here as a tile), and then concatenates them into analysis files. The tiles are removed after writing the analysis files; one may save time for allocating them on disk in the next cycle by leaving them on disk by using option `--leave-tiles`. This approach is somewhat less effective than the direct writing to the analysis files, but, unfortunately, the direct writing is generally not reliable due to parallel I/O issues with NetCDF. Note that in some cases it proved to be possible to obtain robust performance with direct write using "classic" and, possibly, "64-bit" NetCDF formats.

4.7 DA tuning

There are the following main factors for tuning a system on the DA side:

- impact of observations, including the relative impact of different types of observations;
- inflation;
- localisation.

The impact of observations can tuned via the so-called R-factor, which defines the multiple of the observation error variance:

```
RFACTOR BASE = 2
RFACTOR SLA = 1
RFACTOR SST = 2
RFACTOR TEM = 1
RFACTOR SAL = 1
```

Increasing R-factor for observations of some type decreases their impact. The "BASE" R-factor defines a common multiple for observations of all types, while other entries above define additional R-factors for observations of particular types. In the above example the resulting R-factor for SLA, TEM and SAL is 2, while for SST it is 4.

Multiplicative inflation can be seen as an additional forgetting factor in the KF. In EnKF-C one can specify the inflation multiple for analysed ensemble anomalies, e.g.:

```
INFLATION BASE = 1.05
INFLATION temp = 1.02
```

In this case all model variables except "temp" will have inflation of 5 %, while "temp" will have inflation of $1.05 \cdot 1.02 \approx 1.07$. The ability to define different inflation rates for different variables can be useful for non-dynamical variables, such as estimated biases, helping to avoid the ensemble collapse for them. In general, to retain dynamical balances one should rather avoid using different inflation magnitudes. Note that even small inflation can substantially affect the ensemble spread established in the course of evolution of the system.

Localisation radius is defined by the entry `LOCRAD` in the parameter file. Specifically, this entry defines the support localisation radius (in km). This is different to the "effective" localisation radius, which is defined sometimes as $e^{1/2} \approx 1.65$ - folding distance, and for the Gaspary and Cohn's taper function is approximately 3.5 times smaller than the support radius.

Increasing the localisation radius increases the number of local observations and hence the overall impact of observations. To compensate this in a system with horizontal localisation one has to change the R-factor as the square of the localisation radius.

4.8 Point logs

"Point logs" refer to a capability of EnKF-C to save DA related information for a number of horizontal locations specified in the parameter file, e.g.:

```
POINTLOG 94 134
POINTLOG 78 111
POINTLOG 57 51
POINTLOG 86 191
```

Here the information will be saved at each DA cycle for points with horizontal grid coordinates (94, 134), (78, 111), (67, 51), and (86, 191) in files `pointlog_94,134.nc`, `pointlog_78,111.nc`, and so on. The rationale behind this capability is that it is usually too expensive to save the whole ensemble, but is feasible to save the model states in a limited number of points. Following is an example of the header of the saved point log file in NetCDF format:

```
netcdf pointlog_57\,51 {
dimensions:
        p = 2902 ;
        m = 96 ;
        nk = 51 ;
variables:
        int obs_ids(p) ;
        float lcoeffs(p) ;
        float lon(p) ;
        float lat(p) ;
        float depth(p) ;
        float obs_val(p) ;
        float obs_std(p) ;
        float obs_fi(p) ;
        float obs_fj(p) ;
        float obs_fk(p) ;
        int obs_type(p) ;
                obs_type:SST = 0 ;
                obs_type:RFACTOR_SST = 2. ;
                obs_type:SLA = 1 ;
                obs_type:RFACTOR_SLA = 1. ;
                obs_type:TEM = 2 ;
                obs_type:RFACTOR_TEM = 1. ;
                obs_type:SAL = 3 ;
                obs_type:RFACTOR_SAL = 1. ;
        float obs_date(p) ;
                obs_date:units = "days from 8797.5 days since 1990-01-01" ;
        float s(p) ;
        float S(p, m) ;
        double X5(m, m) ;
        float eta_t(m) ;
        float temp(nk, m) ;
        float salt(nk, m) ;
        float u(nk, m) ;
        float v(nk, m) ;

// global attributes:
                :date = "8797.5 days since 1990-01-01" ;
                :i = 57 ;
                :j = 51 ;
                :lon = 95.75 ;
                :lat = -44.8499984741211 ;
                :depth = 3072. ;
```

This information makes it possible to check DA algorithms by reproducing the ensemble transforms calculated by EnKF-C; or to monitor the ensemble spread for each model variable.

4.9 System issues

4.9.1 Memory footprint

To reduce the memory footprint, most of the potentially big arrays in EnKF-C use `float` data type.

The memory footprint of *prep* is defined by the size of the `measurement` structure and the number of observations. It is rarely a problem.

The memory footprint of *calc* is mainly defined by the size of ensemble observation anomalies that require $p \times m \times 4$ bytes for storage using `float` data type. For example, with $3 \cdot 10^6$ superobservations and 10^2 ensemble members the size of this array would be about 1.2 GB per CPU, which should be manageable on most contemporary systems. If the footprint becomes too big, one may consider reducing the number of observations by a coarser superobing (setting the parameter `SOBSTRIDE` to 2 or more) or reducing the number of cores per node used.

The memory footprint of *post* is mainly defined by the size of the array of simultaneously updated horizontal fields. For example, for a 1500×3600 horizontal field, 100 ensemble members and simultaneous update of 2 fields the size of this array would be about 4.3 GB per CPU. It could be reduced to 2.15 GB by reducing he parameter `FIELDBUFFERSIZE` from 2 to 1. Note that reducing the number of simultaneously updated fields defined by `FIELDBUFFERSIZE` increases I/O (the $\mathbf{X}_5$ array is read from disk N_f/N_b times, where N_f is the total number of horizontal fields, and N_b is the number of simultaneously updated fields) and reduces computational effectiveness (the $\mathbf{X}_5$ array needs to be interpolated horizontally N_f/N_b times).

4.9.2 Exit action

When exiting on an error, EnKF-C by default prints the stack trace, which allows to trace the exit location in the code. Another option – to generate a segmentation fault – can be activated by setting `EXITACTION = SEGFAULT` in the parameter file. Note that when run on multiple processors, this can result in segmentation faults on more than one processor (but not necessarilly on every engaged processor, as some processes can also be forced to exit by `MPI_abort()`). If the system is set to generate core dumps, they can indeed be used for investigating the final state of the program.

4.9.3 Dependencies

Compiling EnKF-C requires the following external packages:

- netcdf;
- lapack (or mkl_rt);
- openmpi;
- gridutils.

Notes:

1. The Intel's version of the Lapack library – Intel Math Kernel Library – can improve performance over the Lapack library compiled with gfortran.
2. It should be straightforward to exclude dependence on gridutils if the model does not use curvilinear coordinates.

4.10 Possible problems

4.10.1 *calc* becomes too slow after increasing localisation radius

This is due to the increased number of local observations.

The local observations are sought by using so called k-d tree. Normally it works well, but can become a bottleneck when the number of local observations becomes very large. If this happens, one may use a coarser superobing to reduce the number of assimilated observations, by increasing parameter `SOBSTRIDE` from the default value of 1 to 2 or more.

Acknowledgements

EnKF-C has been developed during author's work with Bureau of Meteorology on Bluelink project. The author used his knowledge of TOPAZ (Sakov et al., 2012) and BODAS (Oke et al., 2008) codes and borrowed from them a number of design solutions and features.

References

Bishop, C. H., B. Etherton, and S. J. Majumdar, 2001: Adaptive sampling with the ensemble transform Kalman filter. part I: theoretical aspects. *Mon. Wea. Rev.*, **129**, 420–436.

Evensen, G., 1994: Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte-Carlo methods to forecast error statistics. *J. Geophys. Res.*, **99**, 10143–10162.

— 2003: The Ensemble Kalman Filter: theoretical formulation and practical implementation. *Ocean Dynam.*, **53**, 343–367.

— 2004: Sampling strategies and square root analysis schemes for the EnKF. *Ocean Dynam.*, **54**, 539–560.

Evensen, G. and P. J. van Leeuwen, 2000: An ensemble Kalman smoother for nonlinear dynamics. *Mon. Wea. Rev.*, **128**, 1852–1867.

Gaspari, G. and S. E. Cohn, 1999: Construction of correlation functions in two and three dimensions. *Q. J. R. Meteorol. Soc.*, **125**, 723–757.

Hamill, T. M. and J. S. Whitaker, 2001: Distance-dependent filtering of background error covariance estimates in an ensemble Kalman filter. *Mon. Wea. Rev.*, **129**, 2776–2790.

Houtekamer, P. L. and H. L. Mitchell, 2001: A sequential ensemble Kalman filter for atmospheric data assimilation. *Mon. Wea. Rev.*, **129**, 123–137.

Hunt, B. R., E. Kalnay, E. J. Kostelich, E. Ott, D. J. Patil, T. Sauer, I. Szunyogh, J. A. Yorke, and A. V. Zimin, 2004: Four-dimensional ensemble Kalman filtering. *Tellus*, **56A**, 273–277.

Hunt, B. R., E. J. Kostelich, and I. Szunyogh, 2007: Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Physica D*, **230**, 112–126.

Kalman, R. E., 1960: A new approach to linear filtering and prediction problems. *J. Basic. Eng.*, **82**, 35–45.

Oke, P. R., G. B. Brassington, D. A. Griffin, and A. Schiller, 2008: The Bluelink ocean data assimilation system (BODAS). *Ocean Model.*, **21**, 46–70.

Ott, E., B. R. Hunt, I. Szunyogh, A. V. Zimin, E. J. Kostelich, M. Corazza, E. Kalnay, D. J. Patil, and J. A. Yorke, 2003, rev. 2005: A local ensemble Kalman filter for atmospheric data assimilation. *http://arxiv.org/abs/physics/0203058*.

Sakov, P. and L. Bertino, 2011: Relation between two common localisation methods for the EnKF. *Comput. Geosci.*, **15**, 225–237.

Sakov, P., F. Counillon, L. Bertino, K. A. Lisæter, P. R. Oke, and A. Korablev, 2012: TOPAZ4: an ocean-sea ice data assimilation system for the North Atlantic and Arctic. *Ocean Science*, **8**, 633–656.

Sakov, P., G. Evensen, and L. Bertino, 2010: Asynchronous data assimilation with the EnKF. *Tellus*, **62A**, 24–29.

Sakov, P. and P. R. Oke, 2008a: A deterministic formulation of the ensemble Kalman filter: an alternative to ensemble square root filters. *Tellus*, **60A**, 361–371.

— 2008b: Implications of the form of the ensemble transformations in the ensemble square root filters. *Mon. Wea. Rev.*, **136**, 1042–1053.

Abbreviations

CL	-	covariance localisation
DEnKF	-	deterministic EnKF
DA	-	data assimilation
DAS	-	data assimilation system
DAW	-	data assimilation window
DFS	-	degrees of freedom of signal
EKF	-	extended Kalman filter
EnKF	-	ensemble Kalman filter
EnOI	-	ensemble optimal interpolation
ETKF	-	ensemble transform Kalman filter
ETM	-	ensemble transform matrix
KF	-	Kalman filter
KS	-	Kalman smoother
LA	-	local analysis
SDAS	-	state of data assimilation system
SRF	-	spread reduction factor
SVD	-	singular value decomposition

Symbols

General symbols

$\mathbf{x}$ (small, bold)	-	a vector
$\mathbf{1}$	-	a vector with all elements equal to 1
$\mathbf{0}$	-	a vector with all elements equal to 0
$\mathbf{A}$ (capital, bold)	-	a matrix
$\mathbf{I}$	-	identity matrix
$\mathbf{A}^{\mathrm{T}}$	-	transposed matrix $\mathbf{A}$
$\mathbf{A}^{1/2}$	-	unique positive definite square root of a positive definite matrix $\mathbf{A}$
$\mathrm{tr}(\mathbf{A})$	-	trace of $\mathbf{A}$
$\mathcal{H} \circ \mathcal{M}(\mathbf{x})$	-	$\mathcal{H}\left[\mathcal{M}(\mathbf{x})\right]$
$\mathbf{A} \circ \mathbf{B}$	-	by-element, or Hadamard, or Schur product of matrices

DA related symbols

m - ensemble size
n - state size
p - number of observations
$\mathbf{A}$ - ensemble anomalies, $\mathbf{A} = \mathbf{E} - \mathbf{x}\mathbf{1}^{\mathrm{T}}$
$\mathbf{E}$ - ensemble
$\mathcal{H}$ - nonlinear observation operator; in linear case – affine observation operator
$\mathbf{G}$ - an intermediate matrix in the EnKF analysis, $\mathbf{G} \equiv (\mathbf{I} + \mathbf{S}^{\mathrm{T}}\mathbf{S})^{-1}\mathbf{S}^{\mathrm{T}} = \mathbf{S}^{\mathrm{T}}(\mathbf{I} + \mathbf{S}\mathbf{S}^{\mathrm{T}})^{-1}$
$\mathbf{H}$ - linearised observation operator, $\mathbf{H} = \nabla\mathcal{H}(\mathbf{x})$
$\mathcal{M}$ - nonlinear model operator; in linear case – affine model operator
$\mathbf{M}$ - linearised model operator, $\mathbf{M} = \nabla\mathcal{M}(\mathbf{x})$
$\mathbf{P}$ - state error covariance estimate
$\mathbf{R}$ - observation error covariance
$\mathbf{S}$ - normalised ensemble observation anomalies, $\mathbf{S} = \mathbf{R}^{-1/2}\mathbf{H}\mathbf{A}/\sqrt{m-1}$
$\mathbf{T}_L$ - left-multiplied ensemble transform matrix, $\mathbf{A}^a = \mathbf{T}_L\mathbf{A}^f$
$\mathbf{T}_R$ - right-multiplied ensemble transform matrix, $\mathbf{A}^a = \mathbf{A}^f\mathbf{T}_R$
$\mathbf{X}_5$ - historic symbol for the full ensemble transform matrix, $\mathbf{E}^a = \mathbf{E}^f\mathbf{X}_5$
$\mathbf{s}$ - normalised innovation, $\mathbf{s} = \mathbf{R}^{-1/2}\left[\mathbf{y} - \mathcal{H}(\mathbf{x}^f)\right]/\sqrt{m-1}$
$\mathbf{x}$ - state estimate
$\mathbf{y}$ - observation vector
$\mathbf{w}$ - vector of linear coefficients for updating the mean, $\mathbf{x}^a = \mathbf{x}^f + \mathbf{A}^f\mathbf{w}$
$(\cdot)^f$ - forecast expression
$(\cdot)^a$ - analysis expression
$(\cdot)_i$ - either expression at cycle i or ith element of a vector
$\overset{i}{(\cdot)}$ - local expression for state element i