Particle filters for high-dimensional geoscience applications: a review

Article

Accepted Version

Van Leeuwen, P. J., Künsch, H. R., Nerger, L., Potthast, R. and Reich, S. (2019) Particle filters for high-dimensional geoscience applications: a review. Quarterly Journal of the Royal Meteorological Society, 145 (723). pp. 2335-2365. ISSN 1477-870X doi: https://doi.org/10.1002/qj.3551 Available at https://centaur.reading.ac.uk/83862/

It is advisable to refer to the publisher's version if you intend to cite from the work. See Guidance on citing.

To link to this article DOI: http://dx.doi.org/10.1002/qj.3551

Publisher: Royal Meteorological Society

All outputs in CentAUR are protected by Intellectual Property Rights law, including copyright law. Copyright and IPR is retained by the creators or other copyright holders. Terms and conditions for use of this material are defined in the End User Agreement.

www.reading.ac.uk/centaur

CentAUR

Central Archive at the University of Reading

University of Reading

Reading's research outputs online

Particle filters for high-dimensional geoscience applications: A Review [†]

Peter Jan van Leeuwen[a,b*], Hans R. Künsch[c],Lars Nerger[d], Roland Potthast[e], Sebastian Reich[f]

[a]*Department of Meteorology and National Centre for Earth Observation, University of Reading, Reading, UK*

[b]*Department of Atmospheric Sciences, Colorado State University, USA*

[c]*Seminar für Statistik, ETH Zurich, Swiszerland*

[d]*Alfred Wegener Institute Helmholtz Center for Polar and Marine Research, Bremerhaven, Germany*

[e] *Deutscher Wetterdienst, Offenbach, Germany*

[f]*Institut für Mathematik, Universitaet Potsdam, Potsdam, Germany*

[*]Correspondence to: Peter Jan van Leeuwen, Department of Meteorology, University of Reading, Reading RG6 6BB, UK. E-mail: p.j.vanleeuwen@reading.ac.uk

Particle filters contain the promise of fully nonlinear data assimilation. They have been applied in numerous science areas, including the geosciences, but their application to high-dimensional geoscience systems has been limited due to their inefficiency in high-dimensional systems in standard settings. However, huge progress has been made, and this limitation is disappearing fast due to recent developments in proposal densities, the use of ideas from (optimal) transportation, the use of localisation and intelligent adaptive resampling strategies. Furthermore, powerful hybrids between particle filters and ensemble Kalman filters and variational methods have been developed. We present a state of the art discussion of present efforts of developing particle filters for high-dimensional nonlinear geoscience state-estimation problems with an emphasis on atmospheric and oceanic applications, including many new ideas, derivations, and unifications, highlighting hidden connections, including pseudo code, and generating a valuable tool and guide for the community. Initial experiments show that particle filters can be competitive with present-day methods for numerical weather prediction suggesting that they will become mainstream soon.

Key Words: nonlinear data assimilation, particle filters; proposal densities, localisation, hybrids

Received . . .

1. Introduction

Data assimilation for geoscience applications, such as weather or ocean prediction, is a slowly maturing field. Even the linear data-assimilation problem cannot be solved adequately because of the size of the problem. Typically, global-scale numerical weather prediction needs estimation of over 10^9 state variables, assimilating over 10^7 observations every 6-12 hours. Existing methods like 4DVar do not provide accurate uncertainty estimates and need efficient pre-conditioners, while Ensemble Kalman Filters heavily rely on somewhat ad-hoc fixes like localisation and inflation to find accurate estimates. Hybrids of variational and ensemble Kalman filter methods are a step forward, although localisation and inflation are still needed in realistic applications. An extra complication is localisation over time needed in ensemble smoothers like the Ensemble Kalman Smoother and 4DEnsVar when the fluid flow is strong: what is local at observation time is not necessary local at the start of the assimilation window because the observation influence is advected with the flow. Furthermore, the recent surge of papers on accurate treatment of observation errors shows that a long way is still ahead of us to solve even the (close to) linear data-assimilation problem.

Although these problems are formidable, another difficulty arises from the fact that the problem is typically nonlinear, and, with increasing model resolution and more complex observation operators, increasingly so. Both variational and Kalman-filter-like methods have difficulty handling nonlinear problems. Variational methods can easily fail when the cost function is multimodal, and are hampered by the assumption that the prior probability density function (pdf) of the state is assumed to be Gaussian. Ensemble Kalman filters make the explicit assumption that the prior pdf and the likelihood of the observations as function of the state are Gaussian, or, somewhat equivalently, assume that the analysis is a linear combination of prior state and observations. Both methods have been shown to fail for nonlinear data-assimilation problems in low-dimensional systems, and both have been reported to have serious difficulties in numerical weather prediction at the convective scale where the model resolution is only a few km. Particle filters hold the promise of fully nonlinear data assimilation without any assumption on prior or likelihood, and recent text books like Reich and Cotter (2015), Nakamura and Potthast (2015), and van Leeuwen *et al.* (2015) provide useful introductions to data-assimilation in general, and particle filters in particular.

Other fully nonlinear data-assimilation methods are Markov-Chain Monte-Carlo methods that draw directly from the posterior in a sequential way, so one sample after the other, after a burn-in period, see e.g. Robert and Cassela (2004), or van Leeuwen *et al.* (2015) for a geophysics-friendly introduction. The samples are correlated, often 100% when the new sample is not accepted, making them very inefficient in high-dimensional systems. This is why we concentrate on particle filtering here.

The standard or bootstrap particle filter can be described as follows. The starting point is an ensemble of size N of model states $\mathbf{x}_i^n \in \Re^{N_x}$, called particles, that represent the prior probability density function (pdf) $p(\mathbf{x}^n)$, as:

$$p(\mathbf{x}^n) \approx \sum_{i=1}^{N} \frac{1}{N}\delta(\mathbf{x}^n - \mathbf{x}_i^n) \tag{1}$$

Between observations, each of these particles is propagated forward from time $n-1$ to time n with the typically nonlinear model equations

$$\mathbf{x}^n = f(\mathbf{x}^{n-1}) + \boldsymbol{\beta}^n \tag{2}$$

in which $f(..)$ denotes the deterministic model, and $\boldsymbol{\beta}^n$ is a random forcing representing missing physics, discretisation errors, etc. In this paper we assume this model noise to be additive, but one could also consider multiplicative noise in which $\boldsymbol{\beta}^n$ is a function of the state of the system. We assume that the pdf from which the $\boldsymbol{\beta}^n$ are drawn is known; typically a Gaussian $N(0,\mathbf{Q})$.

At observation times the true system is observed via:

$$\mathbf{y}^n = \mathbf{H}(\mathbf{x}_{true}^n) + \boldsymbol{\epsilon}^n \tag{3}$$

in which the observation errors $\boldsymbol{\epsilon}^n$ are random vectors representing measurement errors and possibly representation errors. Again we assume that these errors have known

[†]Please ensure that you use the most up to date class file, available from the QJRMS Home Page at
http://onlinelibrary.wiley.com/journal/10.1002/(ISSN)1477-870X

characteristics, often Gaussian, so e.g. $\boldsymbol{\epsilon}^n \sim N(0, \mathbf{R})$. These observations $\mathbf{y}^n \in \Re^{N_y}$ are assimilated by multiplying the prior pdf above with the likelihood of each possible state, i.e. the probability density $p(\mathbf{y}^n|\mathbf{x}^n)$ of the observation vector given each possible model state, following Bayes Theorem:

$$p(\mathbf{x}^n|\mathbf{y}^n) = \frac{p(\mathbf{y}^n|\mathbf{x}^n)}{p(\mathbf{y}^n)} p(\mathbf{x}^n) \tag{4}$$

in which $p(\mathbf{x}^n|\mathbf{y}^n)$ is the posterior pdf, the holy grail of data assimilation. To avoid confusion, it is good to realise that the true state is not a random variable when we apply Bayes Theorem. It is a realisation of a process, which could be random or deterministic, from which we then take noisy observations. Instead, Bayes Theorem is a statement of what we think the true state might be. Since the pdf of the $\boldsymbol{\epsilon}^n$ is known and Bayes Theorem is a statement for each possible state $\mathbf{x}^n$ to be the true state, $p(\mathbf{y}^n|\mathbf{x}^n)$ is the pdf of $\mathbf{y}^n$ given that the true state vector would be $\mathbf{x}^n$. In general, since for a given state $\mathbf{x}^n$ the observation $\mathbf{y}^n$ is equal to the observation error $\boldsymbol{\epsilon}$ shifted by $\mathbf{H}(\mathbf{x}^n)$, we find (see e.g. van Leeuwen (2015)):

$$p(\mathbf{y}^n|\mathbf{x}^n) = p_{\boldsymbol{\epsilon}}(\mathbf{y}^n - \mathbf{H}(\mathbf{x}^n)) \tag{5}$$

If we insert our particle representation of the prior into this theorem we find:

$$p(\mathbf{x}^n|\mathbf{y}^n) \approx \sum_{i=1}^{N} w_i \delta(\mathbf{x}^n - \mathbf{x}_i^n) \tag{6}$$

in which the particle weights w_i are given by:

$$w_i^n = \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{N p(\mathbf{y}^n)} = \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{N \int p(\mathbf{y}^n|\mathbf{x}^n) p(\mathbf{x}^n)\, d\mathbf{x}^n} \approx \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{\sum_j p(\mathbf{y}^n|\mathbf{x}_j^n)} \tag{7}$$

Since all terms are known explicitly we can just calculate this as a number. The self-normalisation in the last part of (7) is consistent with the notion that for a proper representation of a pdf the sum of the weights should be equal to one, so that the integral over the whole state space of the particle representation of the pdf is equal to one. Figure 1 depicts the working of this filter.

Propagating the particles $\mathbf{x}_i^n$ to the next observation time $n+1$ gives a weighted representation of the prior at time $n+1$. Assimilating the observation at time $n+1$ by Bayes Theorem

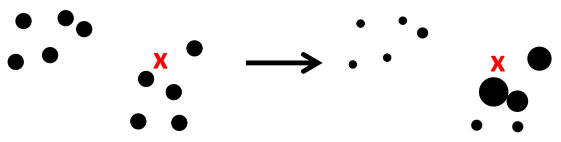

Figure 1. The standard particle filter. Left: the prior particles (dots), with one observation, denoted with the red cross. Right: the posterior particles, the larger the dot the larger its weight. Note that the particles don't move in state space, they are just reweighted.

leads to a modification of the weights (see e.g. Doucet *et al.* (2001) or van Leeuwen (2009)):

$$w_i^{n+1} = w_i^n \frac{p(\mathbf{y}^{n+1}|\mathbf{x}_i^{n+1})}{\sum_j p(\mathbf{y}^{n+1}|\mathbf{x}_j^{n+1})} \tag{8}$$

Even in low-dimensional applications, the variation of the weights increases with the number of assimilation steps. Eventually one particle has a much higher weight than all the others. To prevent this, resampling can be used before propagation to obtain equally weighted particles. This duplicates high-weight particles and abandons low-weight particles. After resampling, some of the particles have identical values, but if the model contains a stochastic component and independent random forcings are used for different particles, diversity is restored. See e.g. Doucet *et al.* (2001) or van Leeuwen (2009) for details. Algorithm 1 illustrates the steps.

Algorithm 1 Standard Particle Filter

```
for $i = 1, .., N$ do
    $w_i \leftarrow p(\mathbf{y}|\mathbf{x}_i^n)$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
Resample
```

A simple resampling scheme using only one draw from a uniform distribution U is presented in Algorithm 2.

In high-dimensional problems the weights vary enormously even at one observation time, and typically one particle obtains a much higher weight than all the others. Snyder *et al.* (2008, 2015) have shown that the number of particles needed to avoid weights collapse, in which one particle gets weight 1 and the rest weights very close to zero, has to grow exponentially with the dimension of the observations $\mathbf{y}$ for a large class of particle filters. If the weights collapse, all particles are identical after resampling, and

Algorithm 2 Simple Resampling Scheme

```
$\hat{w}_1 \leftarrow w_1$
for $j = 2, .., N$ do
    $\hat{w}_j = \sum_{i=1}^{j} w_j$
end for
$u \sim U[0, 1/N]$
$m \leftarrow 1$
for $j = 2, .., N$ do
    while $u > \hat{w}_m$ do
        $m \leftarrow m + 1$
        $\mathbf{x}_m^{new} = \mathbf{x}_j$
    end while
    $u \leftarrow u + 1/N$
end for
```

all diversity is lost. From this discussion it becomes clear that for particle filters to work we need to ensure that their weights remain similar.

In this review we will discuss four basic ways to make progress on this fundamental problem of weight degeneracy. In the first one, we explore the so-called proposal-density freedom to steer particles through state space such that they obtain very similar weights, see e.g. Doucet *et al.* (2001). As pointed out by e.g. Snyder *et al.* (2008) there are fundamental problems when applying these techniques to the high-dimensional geoscience applications. We will examine the issue in detail and discuss so-called equal-weight particle filters, which point towards new ways to formulate and attack the degeneracy problem.

The second approach transform the prior particles into particles from the posterior, either in one go, or via a more smooth transformation process, see Reich (2013). While the one-step approaches can be shown to fail in high-dimensional settings, they do lend themselves very naturally to localisation. The more smooth multi-step transition variants seem to be able to avoid the degeneracy problem without localisation, and are an interesting new development.

The third, more straightforward from the geoscience experience, approach is to introduce localisation in particle filters. While initial implementations were discouraging (e.g. Van Leeuwen, 2009), new formulations have shown remarkable successes, such that localised particle filters are now tested in global operational numerical weather prediction systems (e.g. Potthast *et al.* 2019).

The fourth approach is to abandon the idea of using pure particle filters and combine them with Ensemble Kalman Filters. This should not be confused with using Ensemble Kalman Filters in proposal densities. Several variants exist, such as second-order exact filters, in which only the first two moments are estimated, sequential versions in which first an EnKF is used and the posterior EnKF ensemble is used as input for the particle filter, or vice versa, and combinations in which localised weights are calculated and dependent on the effective ensemble size a full particle filter, an EnKF, or a combination of both is used.

These four variants form the basis of the following four chapters. Each chapter contains a critical discussion of the approximations and remaining major issues. It should be noted that the pseudo code provided does not give the most efficient implementation of the different particle filters, but is rather an illustration of the computational steps involved. Efficient pseudo code for some of the more complex schemes can be found in Vetra-Carvalho *et al.* (2018). The paper is closed with a concluding section and an outlook of what possible next steps could be.

2. Proposal density particle filters

Ideally we draw independent samples directly from the posterior pdf because the samples would all have equal weight automatically. This can only be done, however, when the shape of the posterior pdf is known and when it is easy to draw from the posterior. An example of this is a Gaussian prior combined with a linear Gaussian likelihood. Under these assumptions the posterior is also Gaussian and the mean and covariance can be calculated directly from the prior using the Kalman update equations. Ensemble Kalman filters make use of this result and draw directly from that pdf, which is why all posterior particles have equal weights in an Ensemble Kalman Filter.

The standard particle filter draws particles from the prior. These then have to be modified to become particles of the posterior via the weighting with the likelihood. This is a general procedure in statistics called importance sampling: one draws from an approximation of the pdf one is interested in, and corrects for this via so-called importance weights.

In the introduction we argued that drawing from the prior leads to weights that vary too much: typically, in high-dimensional problems with numerous independent observations one particle gets weight 1, and all other particles have a weight very close to zero. However, we could explore the idea of importance sampling on the transition from one time to the next. When the numerical model is not deterministic but stochastic we have the freedom to change the model equations to move the particles to those parts of state space where we want them to be, for instance closer to the observations.

Mathematically this works as follows. Assume we have observations at time n, so Bayes Theorem at time n is given by (4). If the model is stochastic, we can write the prior as

$$p(\mathbf{x}^n) = \int p(\mathbf{x}^n|\mathbf{x}^{n-1})p(\mathbf{x}^{n-1})\, d\mathbf{x}^{n-1} \tag{9}$$

where $p(\mathbf{x}^n|\mathbf{x}^{n-1})$ is the transition density, the pdf of the state at time n when the state at time $n-1$ is known. For instance, if the model error is additive and the model equation is given by (2), it holds that

$$p(\mathbf{x}^n|\mathbf{x}^{n-1}) = p_{\boldsymbol{\beta}}\left(\mathbf{x}^n - f(\mathbf{x}^{n-1})\right). \tag{10}$$

Often the model errors are assumed to be Gaussian $\boldsymbol{\beta} \sim N(0, \mathbf{Q})$, and we find

$$p(\mathbf{x}^n|\mathbf{x}^{n-1}) = N(f(\mathbf{x}^{n-1}), \mathbf{Q}). \tag{11}$$

but the method is more general than that.

Assume now that at time $n-1$ we have a set of weighted particles as in (1), but with weights w_i^{n-1} instead of $1/N$. We can evaluate the expression (9) for the prior as a weighted mixture of transition densities

$$p(\mathbf{x}^n) \approx \sum_{i=1}^N w_i^{n-1} p(\mathbf{x}^n|\mathbf{x}_i^{n-1}) \tag{12}$$

In the following we neglect the approximation error at time $n-1$ and assume that (12) is exact. This is not necessarily a good approximation, especially when the number of particles is small. On the other hand, it is consistent with the particle filter approximation in the first place, and one of the few things one can do. By Bayes formula (4), the posterior can then be written as:

$$p(\mathbf{x}^n|\mathbf{y}^n) \approx \sum_{i=1}^N w_i^{n-1} \frac{p(\mathbf{y}^n|\mathbf{x}^n)}{p(\mathbf{y}^n)} p(\mathbf{x}^n|\mathbf{x}_i^{n-1}) \tag{13}$$

In the standard particle filter one makes one draw from $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$ for each i, and we know that this leads to ensemble collapse for high-dimensional systems. However, now the prior particles at time n are allowed to arise from following a different model equation. This works as follows. We can multiply and divide equations (12) and (13) by a so-called proposal density $q(\mathbf{x}^n|\mathbf{x}^{n-1}, \mathbf{y}^n)$, leading to:

$$p(\mathbf{x}^n) \approx \sum_{i=1}^N w_i^{n-1} \frac{p(\mathbf{x}^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)} q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) \tag{14}$$

and

$$p(\mathbf{x}^n|\mathbf{y}^n) \approx \sum_{i=1}^N w_i^{n-1} \frac{p(\mathbf{y}^n|\mathbf{x}^n)}{p(\mathbf{y}^n)} \frac{p(\mathbf{x}^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)} q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) \tag{15}$$

where $q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)$ should be non-zero whenever $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$ is. This step is completely general.

Now realise that drawing from $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$ corresponds to running the original stochastic model. We could instead draw from $q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)$, which would correspond to a model equation from our choosing. Figure 2 illustrates the basic idea.

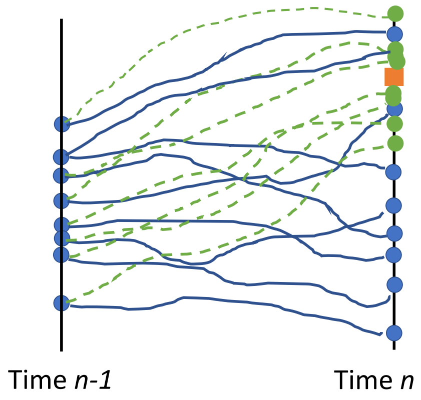

Figure 2. The proposal density. At time $n-1$ we have a set of particles denoted by the filled circles. When we use the original model, they are propagated along the blue lines to time n. Because their distance to the observation (the box) varies significantly, so will their weights. When a proposed model is used the particles at time $n-1$ propagate along the green dashed lines and end up much closer to the observations. This leads to much more similar likelihood weights. However, because we have changed the model equations the particles now also have proposal weights.

For instance when the original model is given by (2), we can use

$$\mathbf{x}^n = g(\mathbf{x}^{n-1}, \mathbf{y}^n) + \hat{\boldsymbol{\beta}}^n \tag{16}$$

in which $g(.,.)$ is now the deterministic part and $\hat{\beta}^n$ is the stochastic part. These can be freely chosen, and examples of these will be given below. Note that we allowed $g(..)$ to depend on the observations at the future time. This means that we generate the prior particles at time n by making one draw from $q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$ for each i where

$$q(\mathbf{x}^n|\mathbf{x}^{n-1},\mathbf{y}^n) = p_{\hat{\beta}}\left(\mathbf{x}^n - g(\mathbf{x}^{n-1},\mathbf{y}^n)\right) \tag{17}$$

In general, we draw the particles at time n from the alternative model $q(\mathbf{x}^n|\mathbf{x}^{n-1},\mathbf{y}^n)$ and account for this by changing the weights of the particles. Equations (14) and (15) can be written as

$$p(\mathbf{x}^n) = \sum_{i=1}^{N} \hat{w}_i^{n-1} q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n) \tag{18}$$

and

$$p(\mathbf{x}^n|\mathbf{y}^n) = \sum_{i=1}^{N} \hat{w}_i^{n} q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n) \tag{19}$$

where the weights are given by:

$$\hat{w}_i^{n-1} \propto w_i^{n-1} \frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}_i^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)}. \tag{20}$$

and

$$\hat{w}_i^n \propto \hat{w}_i^{n-1} \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{p(\mathbf{y}^n)} \propto w_i^{n-1} p(\mathbf{y}^n|\mathbf{x}_i^n) \frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}_i^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)}. \tag{21}$$

Here the coefficients of proportionality ensure that the weights sum to 1. In a reinterpretation of these equations, if $\mathbf{x}_i^n$ is drawn from the alternative model $q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$ we can also write

$$p(\mathbf{x}^n) \approx \sum_{i=1}^{N} \hat{w}_i^{n-1} \delta(\mathbf{x}^n - \mathbf{x}_i^n) \tag{22}$$

and

$$p(\mathbf{x}^n|\mathbf{y}^n) \approx \sum_{i=1}^{N} \hat{w}_i^{n} \delta(\mathbf{x}^n - \mathbf{x}_i^n). \tag{23}$$

We see that the weights now contain two factors, the likelihood weight, which also appears in the standard particle filter, and a proposal weight. These two weights have opposing effects. If we use a proposal density that strongly pushes the model towards the observations, the likelihood weight will be large because the difference between observations and model states becomes smaller, but the proposal weight becomes smaller because the model is pushed away from where it wants to go, so $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$ will be small. On the other hand, a weak pushing towards the observations keeps the proposal weight high, but leads to a small likelihood weight. This suggests that there is an optimum weight related to an optimal position $\mathbf{x}_i^n$ for each particle as function of its position at time $n-1$. This will be explored in equal-weight formulations of the particle filter. Figure 3 shows how typical proposal-density particle filters work. Equal-weight particle filters are discussed later.

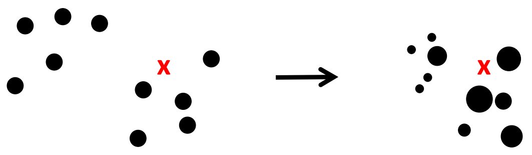

Figure 3. The typical proposal-density particle filter. Left: the prior particles at time $n-1$ (dots), with one observation, denoted with the red cross. Right: the posterior particles at time n, the larger the dot the larger its weight. Note that the particles do move in state space compared to a pure model propagation over one time step, and their weight contains contributions from the likelihood and from that movement.

2.1. A simple relaxation scheme

To illustrate the idea of a proposal density we consider the following simple example. We could add a relaxation or nudging term to the original equation to steer the particles towards the observations and make their weights more similar, as pioneered by van Leeuwen (2010) for geoscience applications. The model equation is written as:

$$\mathbf{x}^m = f(\mathbf{x}^{m-1}) + \mathbf{T}(\mathbf{y}^n - \mathbf{H}(\mathbf{x}^{m-1})) + \hat{\beta}^m \tag{24}$$

where we used time index m for the state vector to emphasise that there are several model time steps between observation times. $\mathbf{T}$ is a relaxation matrix of our choice. In this example, the deterministic part consists of the first two terms on the right-hand side of the equation, while the third term denotes the random part. Let's assume the pdf of the random forcing is Gaussian with mean zero and covariance $\hat{\mathbf{Q}}$. Then we can immediately write for the proposal density

$$q(\mathbf{x}^m|\mathbf{x}^{m-1},\mathbf{y}^n) = N\left(f(\mathbf{x}^{m-1}) + \mathbf{T}(\mathbf{y}^n - \mathbf{H}(\mathbf{x}^{m-1})), \hat{\mathbf{Q}}\right) \tag{25}$$

since the pdf of $\mathbf{x}^m$ is just a shift in the mean of the pdf of $\hat{\boldsymbol{\beta}}^m$. For the original model, we assume that the random part is Gaussian with zero mean and covariance $\mathbf{Q}$, so that

$$p(\mathbf{x}^m|\mathbf{x}^{m-1}) = N\left(f(\mathbf{x}^{m-1}), \mathbf{Q}\right) \tag{26}$$

The change in the model equations is compensated for in particle filters by a change in the relative weight of each particle, and the expression for this change in weight for this case is:

$$\begin{aligned} w_i^m &= w_i^{m-1}\frac{p(\mathbf{x}_i^m|\mathbf{x}_i^{m-1})}{q(\mathbf{x}_i^m|\mathbf{x}_i^{m-1},\mathbf{y}^n)} \\ &\propto w_i^{m-1}\frac{\exp\left[-J_p\right]}{\exp\left[-J_q\right]} \end{aligned} \tag{27}$$

in which, for Gaussian model errors,

$$J_p = \frac{1}{2}\left(\mathbf{x}_i^m - f(\mathbf{x}_i^{m-1})\right)^T \mathbf{Q}^{-1}\left(\mathbf{x}_i^m - f(\mathbf{x}_i^{m-1})\right) \tag{28}$$

and

$$\begin{aligned} J_q &= \frac{1}{2}\left(\mathbf{x}_i^m - f(\mathbf{x}_i^{m-1}) - \mathbf{T}(\mathbf{y}^n - H(\mathbf{x}^{m-1}))\right)^T \cdot \\ &\quad \hat{\mathbf{Q}}^{-1}\left(\mathbf{x}_i^m - f(\mathbf{x}_i^{m-1}) - \mathbf{T}(\mathbf{y}^n - \mathbf{H}(\mathbf{x}^{m-1}))\right) \\ &= \frac{1}{2}(\hat{\boldsymbol{\beta}}_i^m)^T\hat{\mathbf{Q}}^{-1}\hat{\boldsymbol{\beta}}_i^m \end{aligned} \tag{29}$$

Note that the normalisation factors of the Gaussians do not have to be calculated explicitly if we use that the sum of the weights has to be equal to one. The scheme is depicted by Algorithm 3.

Algorithm 3 Relaxation Proposal Density

```
for $j = 1, ..., N$ do
    $\mathbf{d}_j \leftarrow \mathbf{y} - H\left(\mathbf{x}_j^f\right)$
    $\mathbf{f}_j \leftarrow \mathbf{T}\mathbf{d}_j$
    $\boldsymbol{\xi}_j \sim N(0, \mathbf{Q})$
    $\mathbf{x}_j^m \leftarrow f\left(\mathbf{x}_j^{m-1}\right) + \mathbf{f}_j + \boldsymbol{\xi}_j$
    $\log w_j^m \leftarrow \log w_j^{m-1} + \frac{1}{2}\boldsymbol{\xi}_j\hat{\mathbf{Q}}^{-1}\boldsymbol{\xi}_j$
    $\log w_j^m \leftarrow \log w_j^m - \frac{1}{2}(\mathbf{f}_j + \boldsymbol{\xi}_j)^T\mathbf{Q}^{-1}(\mathbf{f}_j + \boldsymbol{\xi}_j)$
end for
```

Simple as the scheme is, it does not solve the degeneracy problem. However, it can be used as a simple scheme when several model time steps are used between observation times, because the proposal is independent of the proposal at other time steps. This scheme an easily be used in combination with other schemes that work at observation time, to be discussed next.

2.2. *Weighted Ensemble Kalman Filter*

One could also use other existing data-assimilation methods in proposal densities, like Ensemble Kalman filters or variational methods. In the Weighted Ensemble Kalman filter (Papadakis *et al.* 2010) the stochastic EnKF of Burgers *et al.* (1998) is used as follows. The Ensemble Kalman Filter update can be written as:

$$\mathbf{x}_i^n = \mathbf{x}_i^f + \mathbf{K}(\mathbf{y}^n - \mathbf{H}\mathbf{x}_i^f - \boldsymbol{\epsilon}_i) \tag{30}$$

in which $\mathbf{x}_i^f = f(\mathbf{x}_i^{n-1}) + \boldsymbol{\beta}_i^n$, the matrix K is the ensemble Kalman gain and $\boldsymbol{\epsilon}_i \sim N(0,\mathbf{R})$, with $\mathbf{R}$ the observational error covariance. Using the expression for the forecast $\mathbf{x}_i^f$ in the Kalman filter update equation we find:

$$\mathbf{x}_i^n = f(\mathbf{x}_i^{n-1}) + \mathbf{K}\left(\mathbf{y}^n - \mathbf{H}f(\mathbf{x}_i^{n-1})\right) + (\mathbf{I}-\mathbf{KH})\boldsymbol{\beta}_i^n - \mathbf{K}\boldsymbol{\epsilon}_i \tag{31}$$

which we can rewrite as the sum of a deterministic and a stochastic part as:

$$\mathbf{x}^n = g(\mathbf{x}^{n-1},\mathbf{y}^n) + \hat{\boldsymbol{\beta}}_i^n \tag{32}$$

identifying $g(\mathbf{x}^{n-1}) = f(\mathbf{x}_i^{n-1}) + \mathbf{K}\left(\mathbf{y}^n - \mathbf{H}f(\mathbf{x}_i^{n-1})\right)$ and $\hat{\boldsymbol{\beta}}_i^n = (\mathbf{I}-\mathbf{KH})\boldsymbol{\beta}_i^n - \mathbf{K}\boldsymbol{\epsilon}_i$. Therefore, we find for the proposal density:

$$q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n) = N\left(f(\mathbf{x}^{n-1}) + \mathbf{K}(\mathbf{y}^n - \mathbf{H}f(\mathbf{x}^{n-1}), \hat{\mathbf{Q}}\right) \tag{33}$$

with

$$\hat{\mathbf{Q}} = (\mathbf{I}-\mathbf{KH})\mathbf{Q}(\mathbf{I}-\mathbf{KH})^T + \mathbf{KRK}^T. \tag{34}$$

Strictly speaking, this is correct only if the Kalman gain is calculated using the ensemble covariance of $f(\mathbf{x}^{n-1})$, so without the model errors $\boldsymbol{\beta}^n$, otherwise the proposal is not Gaussian. We can calculate the weights of the particles in a similar way as in the previous example. Algorithm 4 shows the algorithmic steps.

The behaviour of this filter has been studied extensively in Morzfeld *et al.* (2017). In high-dimensional systems this filter will

Algorithm 4 WEKF

```
$\hat{\mathbf{Q}} \leftarrow (\mathbf{I}-\mathbf{KH})\mathbf{Q}(\mathbf{I}-\mathbf{KH})^T + \mathbf{KRK}^T$
for $i = 1, ..., N$ do
    $\hat{\boldsymbol{\beta}}_i \sim N(0, \hat{\mathbf{Q}})$
    $\mathbf{x}_i^n \leftarrow f(\mathbf{x}_i^{n-1}) + \mathbf{K}\left(\mathbf{y}^n - \mathbf{H}f(\mathbf{x}_i^{n-1})\right) + \hat{\boldsymbol{\beta}}_i^n$
    $w_i \leftarrow \frac{1}{2}\left(\mathbf{x}_i^n - f(\mathbf{x}_i^{n-1})\right)\mathbf{Q}^{-1}\left(\mathbf{x}_i^n - f(\mathbf{x}_i^{n-1})\right)$
    $w_i \leftarrow w_i + \frac{1}{2}\hat{\boldsymbol{\beta}}_i\hat{\mathbf{Q}}^{-1}\hat{\boldsymbol{\beta}}_i$
    $w_i \leftarrow w_i + \frac{1}{2}(\mathbf{y} - H(\mathbf{x}_i^n))^T\mathbf{R}^{-1}(\mathbf{y} - H(\mathbf{x}_i^n))$
    $w_i \leftarrow \exp[-w_i]$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
Resample
```

be degenerate, consistent with the theory of Snyder *et al.* (2015), and as proven in the next section. The only way to make this work is to include localisation, not only at the EnKF level, but also at the level of the particle filter, see e.g. Morzfeld *et al.* (2017).

2.3. *Optimal proposal density*

In the class of particle filters in which the proposal density of each particle is dependent on only that particle, an optimal proposal density can be derived, as e.g. shown in Doucet *et al.* (2001). They defined optimality as the proposal density that gives a minimal variance of the weights, and Snyder *et al.* (2015) provide an elegant proof of this optimality. In this section we generalise this result and show that the optimal proposal density is optimal even when each particle has its own proposal density which is allowed to depend on all previous particles, so a proposal of the form $q(\mathbf{x}^n|i, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)$.

Snyder *et al.* (2015) concentrate on the case that one is interested in an optimal representation of $p(x^n, x^{n-1}|y^n)$ in a sequential algorithm, so in a sequential smoother. To this end they introduce the random variable

$$w^*(\mathbf{x}^n, \mathbf{x}^{n-1}) = \frac{p(\mathbf{x}^n, \mathbf{x}^{n-1}|\mathbf{y}^n)}{q(\mathbf{x}^n, \mathbf{x}^{n-1}|\mathbf{y}^n)} \tag{35}$$

and determine that proposal density q that minimises the variance in the weights w^*, with the expectation taken over the density from which we draw the particles, so the proposal q.

Here we show that the optimal proposal density is also optimal for the strict filtering case, so when we are interested in minimal variance of the weights at time n only. Specifically, the question is: given the set of particles at $t = n-1$ drawn from $p(\mathbf{x}_{n-1}|\mathbf{y}_{1:n-1})$, which proposal density of the form $q(\mathbf{x}^n|i, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)$ gives minimal variance of the weights at time n?

Using Bayes formula, we can write the expression for the weight of particle i as function of the state at time n as:

$$\begin{aligned} w_i^n = w_i(\mathbf{x}_i^n) &= \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{Np(\mathbf{y}^n)}\frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}_i^n|i, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)} \\ &= \frac{p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}{Np(\mathbf{y}^n)}\frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)}{q(\mathbf{x}_i^n|i, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)} \end{aligned} \tag{36}$$

where we assume, without loss of generality, an equally weighted ensemble at time $n-1$. Note that the second equality follows from Bayes Theorem, as follows:

$$\begin{aligned} p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) &= \frac{p(\mathbf{y}^n|\mathbf{x}_i^n, \mathbf{x}_i^{n-1})}{p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1}) \\ &= \frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1}) \end{aligned} \tag{37}$$

Consider the pair of random variables $(I, \mathbf{X}^n)$ where $Prob(I = i) = \frac{1}{N}$ and, conditionally on $I = i$, $\mathbf{X}^n \sim q(\mathbf{x}^n|i, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)$. Furthermore, define the associated random variable

$$W = w_I(\mathbf{X}^n) = \frac{p(\mathbf{y}^n|\mathbf{x}_I^{n-1})}{Np(\mathbf{y}^n)}\frac{p(\mathbf{X}^n|\mathbf{x}_I^{n-1}, \mathbf{y}^n)}{q(\mathbf{X}^n|I, \mathbf{x}_{1:N}^{n-1}, \mathbf{y}^n)} \tag{38}$$

where

$$p(\mathbf{y}^n) = \frac{1}{N}\sum_{j=1}^{N} p(\mathbf{y}^n|\mathbf{x}_j^{n-1}) \tag{39}$$

In order to find the proposal q that minimizes the variance of W, we use the well-known law of total variance (derived in the appendix for completeness):

$$var_W(W) = var_I(E_{\mathbf{X}^n|I}(W)) + E_I(var_{\mathbf{X}^n|I}(W)). \tag{40}$$

First, we see that, under the proposal q:

$$E_{\mathbf{X}^n|I}(W) = \frac{p(\mathbf{y}^n|\mathbf{x}_I^{n-1})}{Np(\mathbf{y}^n)}\int p(\mathbf{x}^n|\mathbf{x}_I^{n-1}, \mathbf{y}^n)d\mathbf{x}^n = \frac{p(\mathbf{y}^n|\mathbf{x}_I^{n-1})}{Np(\mathbf{y}^n)} \tag{41}$$

is independent of q. Moreover, $E_W(W) = E_I(E_{\mathbf{X}^n|I}(W)) = 1/N$ and thus the first term in $var_W(W)$ is

$$\frac{1}{N}\sum_i \frac{p(\mathbf{y}^n|\mathbf{x}_i^{n-1})^2}{N^2 p(\mathbf{y}^n)^2} - \frac{1}{N^2} = \frac{1}{N}\sum_i\left(\frac{p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}{Np(\mathbf{y}^n)} - \frac{1}{N}\right)^2 \geq 0. \tag{42}$$

For the second term we use that $var_{\mathbf{X}^n|I}(W) \geq 0$ with equality if and only if W is almost surely constant in $\mathbf{X}^n$, that is if and only if

$$\frac{p(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)}{q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n)} = cst(i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n). \quad (43)$$

in which $cst(..)$ is this constant which can depend on other variables than $\mathbf{x}^n$. Because both p and q are densities (in $\mathbf{x}^n$), $cst = 1$. Combining these results, we have a lower bound for $var(W)$ that is determined by the variance of $p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$ over i, with equality if and only if

$$q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n) = p(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n) \quad (44)$$

Note that this is a new result as previous proofs only considered proposal densities of the form $q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$, and we extended it to more general proposal densities of the form $q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n)$.

This remarkable result shows that firstly the optimal proposal density, so $p(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$, does indeed lead to the lowest variance in the weights for the class of particle filters in which the transition density is of the form $q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n)$. Secondly, it shows that we can predict the variance in the weights without doing the actual experiment, for any number of particles, provided we can compute $p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$, and thirdly the weights are *independent of the position of the particles* $\mathbf{x}^n$. Unfortunately, this variance is zero only when the observations are not dependent on the state at time $n-1$, which is never the case in the geosciences.

A simple case where we can compute both the optimal proposal density and the weights $p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$ is when $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$ is given by (11) and the observation operator $H = \mathbf{H}$ is linear. By the same argument that is used to derive the Kalman filter update, we find

$$p(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n) =$$

$$= N\left(f(\mathbf{x}_i^{n-1}) + \mathbf{T}(\mathbf{y}^n - \mathbf{H}f(\mathbf{x}_i^{n-1})), (\mathbf{I}-\mathbf{T}\mathbf{H}^T)\mathbf{Q}\right), \quad (45)$$

where $\mathbf{T} = \mathbf{Q}\mathbf{H}^T(\mathbf{H}\mathbf{Q}\mathbf{H}^T + \mathbf{R})^{-1}$ is the Kalman-like gain with the background covariance $\mathbf{Q}$, and the weights are proportional to:

$$p(\mathbf{y}^n|\mathbf{x}_i^{n-1}) = N(\mathbf{H}f(\mathbf{x}_i^{n-1}), \mathbf{H}\mathbf{Q}\mathbf{H}^T + \mathbf{R}) \quad (46)$$

This shows two things: First, in this special case, the simple relaxation scheme of Section 2.1 is equal to the optimal proposal when the relaxation matrix $\mathbf{T}$ is chosen as above. Second, comparing the weights of the optimal proposal with the weights of the standard filter, they both depend on the squared distance $||\mathbf{y}^n - \mathbf{H}f(\mathbf{x}_i^{n-1})||^2$, and $||\mathbf{y}^n - \mathbf{H}\mathbf{x}_i^n||^2$, respectively, but in the standard particle filter the distance is defined w.r. to $\mathbf{R}$ and in the optimal proposal the distance it is defined is w.r. to $\mathbf{H}\mathbf{Q}\mathbf{H}^T + \mathbf{R}$. Hence the weights with the optimal proposal are more similar, but the improvement is substantial only if $\mathbf{Q}$ is large, and the analysis of weight collapse by Snyder *et al.* (2008) still applies.

One can extend the optimal proposal density idea to more than one time step. Snyder *et al.* (2015) show that the optimal proposal is the proposal of this form with minimal variance in the weights in this case too, which can also easily be seen by applying the above to

$$W = w_i(\mathbf{x}^n) = \frac{p(\mathbf{y}^n|\mathbf{x}_i^{m-1})}{Np(\mathbf{y}^n)} \frac{p(\mathbf{x}^n|\mathbf{x}_i^{m-1},\mathbf{y}^n)}{q(\mathbf{x}^n|\mathbf{x}_i^{m-1},\mathbf{y}^n)}$$

for $m < n$.

Looking back at the filters described in the previous sections we find the following. The relaxation scheme uses a simple proposal density that is of the form $q(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$, so the theory holds, and that proposal will lead to degenerate results. This is indeed the finding of van Leeuwen (2010). The Weighted Ensemble Kalman Filter has a proposal that depends on all particles at time $n-1$ through the Kalman gain $\mathbf{K}$, so the proposal is of the form $q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n)$. Hence also this filter will perform worse than the optimal proposal and hence will be degenerate for high-dimensional systems. This was first explored in detail by Morzfeld *et al.* (2017).

2.4. Implicit Particle filter

The Implicit Particle Filter is an indirect way to draw from the optimal proposal, even over several time steps. Often the assumption is made that the model errors of both original model and proposal density are Gaussian, and the observation operator $\mathbf{H}$ is linear. In this case, a draw from the optimal proposal is a draw from a multivariate Gaussian, and we know how to do that.

However, when $\mathbf{H}$ is nonlinear, or when the proposal is used over several model time steps the density to draw from is not Gaussian anymore. Chorin *et al.* (2010) realised that one could still draw from a Gaussian and then apply a transformation to that draw to find samples from the optimal proposal density. The method is explained here for one time step, but the extension to multiple time steps is straightforward. Figure 4 illustrates the basic idea.

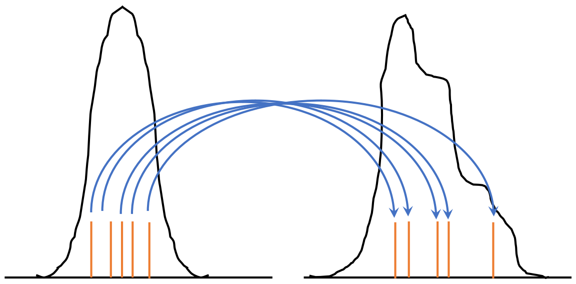
Figure 4. The Implicit Particle Filter. Samples (red bars in left pdf) are drawn from the standard multivariate Gaussian and transformed via equation (49) to weighted samples from the posterior (red bars in right pdf).

As mentioned in Sec. 2 on the proposal density the posterior pdf can be written as:

$$p(\mathbf{x}^n|\mathbf{y}^n) = \sum_{i=1}^{N} w_i^{n-1} \frac{p(\mathbf{y}^n|\mathbf{x}^n)}{p(\mathbf{y}^n)} \frac{p(\mathbf{x}^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)} q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) \tag{47}$$

The scheme draws from a Gaussian proposal $q(\boldsymbol{\xi}) = N(0, \mathbf{I})$, and we can write the transformation as $q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) = q(\boldsymbol{\xi})\mathbf{J}_i^{-1}$ in which $\mathbf{J}_i$ is the Jacobian of the transformation from $\mathbf{x}^n$ to $\boldsymbol{\xi}$. That transformation is found implicitly, hence the name of the filter, by defining

$$F_i(\mathbf{x}^n) = -\log\left[p(\mathbf{y}^n|\mathbf{x}^n)p(\mathbf{x}^n|\mathbf{x}_i^{n-1})\right] \tag{48}$$

and, after drawing $\boldsymbol{\xi}_i$ for each particle, solving for $\mathbf{x}^n$ in

$$F_i(\mathbf{x}^n) = \frac{1}{2}\boldsymbol{\xi}_i^T\boldsymbol{\xi}_i + \phi_i \tag{49}$$

for each particle, in which $\phi_i = \min_{\mathbf{x}^n} F_i(\mathbf{x}^n) \propto p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$. The weights of the particles become:

$$\begin{aligned} w_i^n &= w_i^{n-1}\frac{p(\mathbf{y}^n|\mathbf{x}_i^n)}{p(\mathbf{y}^n)}\frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1})}{q(\mathbf{x}_i^n|\mathbf{x}_i^{n-1}\mathbf{y}^n)} \\ &= w_i^{n-1}\frac{\exp\left[-F_i(\mathbf{x}_i^n)\right]}{\exp\left[-\frac{1}{2}\boldsymbol{\xi}_i^T\boldsymbol{\xi}_i\right]}\mathbf{J}_i \\ &= w_i^{n-1}\frac{\exp\left[-F_i(\mathbf{x}_i^n)\right]}{\exp\left[-F_i(\mathbf{x}^n)+\phi_i\right]}\mathbf{J}_i \\ &= w_i^{n-1}\exp\left[-\phi_i\right]\mathbf{J}_i \end{aligned} \tag{50}$$

Interestingly, while the optimal proposal density shows that the weights are only dependent on the position of the particles at the previous time, so on $\mathbf{x}_i^{n-1}$ via ϕ_i, the implicit map makes the weights also dependent on the positions at the current time n, so on $\mathbf{x}_i^n$ via the Jacobian of the transformation between $\boldsymbol{\xi}$ and $\mathbf{x}$. Only when the Jacobian is a constant, so when F_i is quadratic in $\mathbf{x}_i$, this dependence disappears.

Solving (49) is not straightforward in general. Morzfeld *et al.* (2012) suggest a random map of the form

$$\mathbf{x}_i^n = \mathbf{x}_i^a + \lambda_i(\boldsymbol{\xi}_i)\mathbf{P}^{1/2}\boldsymbol{\xi}_i \tag{51}$$

in which $\mathbf{P}$ is a chosen covariance matrix, ideally the covariance of the posterior pdf, $\mathbf{x}_i^a = \arg\min F_i(\mathbf{x}^n)$ and λ_i is a scalar. This transforms the problem into solving a highly nonlinear scalar equation for λ_i, which is a much simpler problem than finding $\mathbf{x}_i^n$ directly. This map can be shown to be a bijection when $F_i(\mathbf{x}_i^n)$ has only closed contours in the high-probability regions; otherwise one would have to first choose a closed contour area and then perform the map. In general, when the optimal proposal (over several time steps if needed) is multimodal, the transformation from the state variable to a Gaussian is not monotonic, and the Implicit Particle Filter needs to be adapted, e.g. by using a separate Gaussian for each mode. The algorithm is given in Algorithm 5.

Of further interest is that $\mathbf{x}_i^a$ is the same as the solution to a 4DVar problem well known in meteorology. But it is a special 4DVar as the initial position of each particle is fixed and it has to be a weak-constraint 4DVar. The latter condition is needed as a strong-constraint 4Dvar would have no possibility to move a particle in state space as its initial condition is fixed.

Algorithm 5 Implicit Particle Filter

```
for $i = 1, ..., N$ do
    $\boldsymbol{\xi}_i \sim N(0, \mathbf{I})$
    $\phi_i \leftarrow \min_{\mathbf{x}^n}\{-\log\left[p(\mathbf{y}^n|\mathbf{x}^n)p(\mathbf{x}^n|\mathbf{x}_i^{n-1})\right]\}$
    Solve $-\log\left[p(\mathbf{y}^n|\mathbf{x}^n)p(\mathbf{x}^n|\mathbf{x}_i^{n-1})\right] = \frac{1}{2}\boldsymbol{\xi}_i^T\boldsymbol{\xi}_i + \phi_i$ for $\mathbf{x}^n$
    $\mathbf{J}_i = \left|\frac{\partial \mathbf{x}^n}{\partial \boldsymbol{\xi}_i}\right|$
    $w_i \leftarrow \exp\left[-\phi_i\right]\mathbf{J}_i$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
Resample
```

However, also this filter will suffer from weight collapse in high-dimensional applications as it is still a sampling scheme for the optimal proposal density. The following sections will discuss ways to improve on the optimal proposal.

2.5. *Equal weights by resampling at time $n-1$*

As noted already in equation (36), we can write equation (13) as

$$p(\mathbf{x}^n|\mathbf{y}^n) = \sum_{i=1}^{N} \frac{w_i^{n-1}p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}{p(\mathbf{y}^n)} p(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) = \sum_{i=1}^{N} \alpha_i p(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n) \quad (52)$$

where

$$\alpha_i = \frac{w_i^{n-1}p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}{p(\mathbf{y}^n)} \quad (53)$$

This says that, assuming the pdf at the previous time can be approximated by a set of N particles, the analysis distribution is a mixture of the optimal proposal pdf's $p(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)$ with mixture weights α_i.

If we can compute the optimal proposal density and the weights α_i in closed form, we can also draw samples *directly from this mixture density*. For this, we first draw an index I from the discrete distribution with weights α_i, $Prob(I = j) = \alpha_j$, followed by a draw from the corresponding pdf $p(\mathbf{x}^n|\mathbf{x}_I^{n-1}, \mathbf{y}^n)$. Doing this N times will lead to N different particles with equal weights because each of them is an independent draw directly from the posterior. If the index I is equal to a value j more than once, the particle $\mathbf{x}_j^{n-1}$ is propagated from time $n-1$ to time n with independent random forcing for each of these draws. This simple scheme provides better samples than the optimal proposal density because all particles are different at time n by construction.

However, this does not solve the problem of weight collapse because drawing the index I is nothing else than resampling the particles at time $n-1$ with weights proportional to $w_i^{n-1}p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$. If $w_i^{n-1} = \frac{1}{N}$, the variance of these weights is exactly equal to the lower bound that we found in Section 2.3. The main difference is that the collapse now happens at time $n-1$. The only advantage is that all particles will be different at time n.

If we cannot compute the optimal proposal density and the weights α_i in closed form, we can still use the importance sampling idea to draw from the mixture $p(\mathbf{x}^n|\mathbf{y}^n)$ by drawing pairs $(I, \mathbf{X}^n)$ consisting of an index I and a state $\mathbf{X}^n$ at time n. We choose a proposal distribution $\beta_i = \beta_i(\mathbf{y}^n)$ for the index and proposal distributions $q(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)$ for the state. Then we draw the index I_i with $Prob(I_i = j) = \beta_j(\mathbf{y}^n)$ and conditionally on $I_i = j$ we draw $\mathbf{x}_i^n$ from $q(\mathbf{x}^n|\mathbf{x}_j^{n-1}, \mathbf{y}^n)$. Finally, we compute weights w_i^n by

$$w_i^n \propto \frac{w_j^{n-1}p(\mathbf{x}_i^n|\mathbf{x}_j^{n-1})p(\mathbf{y}|\mathbf{x}_i^n)}{\beta_j(\mathbf{y}^n)q(\mathbf{x}^n|\mathbf{x}_j^{n-1}, \mathbf{y}^n)} \text{ if } I_i = j$$

The particles $\mathbf{x}_i^n$ with weights w_i^n provide the desired approximation of $p(\mathbf{x}^n|\mathbf{y}^n)$ whereas the indices I_i can be discarded after the weights have been computed. We could produce an evenly-weighted approximation by a further resampling step, or take the weights w_i^n into account during the next iteration.

In this approach we can obtain equal weights w_i^n by choosing

$$q(\mathbf{x}^n|\mathbf{x}_j^{n-1}, \mathbf{y}^n) = p(\mathbf{x}^n|\mathbf{x}_j^{n-1}, \mathbf{y}^n)$$

and

$$\beta_i(\mathbf{y}^n) \propto w_i^{n-1}p(\mathbf{y}^n|\mathbf{x}_i^{n-1}).$$

With this choice, we draw directly from the mixture (52). As mentioned before, although the weights w_i^n are then equal to $\frac{1}{N}$, the algorithm contains a hidden weighting and resampling step of particles at time $n-1$. It thus remains susceptible to weight collapse in high dimensions.

This approach of using importance sampling for the joint distribution of $(I, \mathbf{X}^n)$ is due to Pitt and Shephard (1999) who called it "Auxiliary Particle Filter" (the index I is an auxiliary variable that is discarded at the end). They discuss, in addition, approximations of the optimal proposal density and the optimal weights α_i. One of their suggestions is to use for the index I the proposal with weights

$$\beta_i \propto w_i^{n-1} p(\mathbf{y}^n|\boldsymbol{\mu}_i^n)$$

where $\boldsymbol{\mu}_i^n$ is a likely value of the distribution $p(\mathbf{x}^n|\mathbf{x}_i^{n-1})$, e.g. the mean or median or simply a draw from it. Typically, $\boldsymbol{\mu}_i^n$ is found by a probing step where particles at time n are propagated by a simplified model, e.g. by omitting stochastic terms or with simplified subgrid-scale parameterisations or thermodynamics. If $I_i = j$ and the state $\mathbf{x}_i^n$ at time n is proposed from $p(\mathbf{x}^n|\mathbf{x}_j^{n-1})$, the weights become

$$w_i^n \propto \frac{p(\mathbf{y}|\mathbf{x}_i^n)}{p(\mathbf{y}^n|\boldsymbol{\mu}_j^n)}$$

They will vary less provided $\mathbf{x}_i^n$ is close to $\boldsymbol{\mu}_j^n$, i.e. provided the simplified model is a good approximation to the full model and the stochastic part of the full model is small.

2.6. The Equivalent-Weights Particle Filter (EWPF)

The EWPF (van Leeuwen 2010; Ades and van Leeuwen 2013) uses the idea to obtain a more evenly weighted set of particles by not sampling from the exact posterior, but allowing for a small error. It starts with determining the weight of each particle at the mode of $p(\mathbf{x}^n|\mathbf{x}_i^{n-1}, \mathbf{y}^n)$ for each particle i, $w_i^{max} \propto p(\mathbf{y}^n|\mathbf{x}_i^{n-1})$. Note that these weights are equal to the weights obtained in the optimal proposal density. In the optimal proposal density case the weights do not depend on the position $\mathbf{x}^n$ of the particle, but note that the proposal used here will be different.

The particles are not moved to these modes, but the weights are used to define a target weight. This target weight w_{target} is chosen such that a certain fraction ρ of particles can reach that weight. To this end we sort the weights in magnitude from high to low in an array w_i^*, $i = \{1, 2, ..., N]\}$ and set $w_{target} = w_{N*\rho}^*$. For instance, with 100 particles and a fraction of $\rho = 0.8$ we would find $w_{target} = w_{80}^*$.

The next step is to find a position in state space for each particle that can reach this weight such that its weight is exactly equal to the target weight. This means we solve for $\mathbf{x}^n$ in

$$w_i(\mathbf{x}^n) = w_{target} \tag{54}$$

for each particle i that can reach this weight. There are many solutions of this equation, but we choose the one which is on the line through $\mathbf{x}_i^a$ and $f(\mathbf{x}_i^{n-1})$ and is closest to $f(\mathbf{x}_i^{n-1})$. Denote this position as $\mathbf{x}_i^*$. Note that this is purely deterministic move, so a stochastic part still has to be added. The final position of these particles is then determined by adding a very small random perturbation $\boldsymbol{\xi}$ from a chosen density, so

$$\mathbf{x}_i^n = \mathbf{x}_i^* + \boldsymbol{\xi}_i^n \tag{55}$$

This stochastic move ensures that the proposal has full support and is not a delta function centred at $\mathbf{x}_i^*$. The density of $\boldsymbol{\xi}_i$ should on the one hand have most of its mass concentrated around 0 in order not to change the weights of the particles too much, and on the other hand it should be relatively constant since we divide by the value of the proposal density. Both requirements cannot be fulfilled exactly, but we can take some error in the sampling into account and choose a narrow uniform distribution. The scheme is depicted in Algorithm 6 for the special case that of Gaussian model errors and a linear observation operator. If these conditions do not hold, one will typically need iterations to solve for a_i and b_i.

It is common knowledge, see e.g. Doucet *et al.* (2001), that the proposal should be wider or at least as wide as the target, while the width of the stochastic part of the proposal is chosen very small here. The reason that we can do this is that the position of the centres of these proposal densities are typically further away from the observations than e.g. in the optimal proposal because the target weight forces particles away from their optimal positions, so away from the observations. This means that the deterministic moves of the particles ensure a large spread in the full proposal.

A formal way to avoid such an error has been described by Ades and van Leeuwen (2015b). They choose the proposal to be a mixture of a uniform density and a Gaussian which is also used in Alg. 6. Both have small variance, and the mixture coefficient of the uniform density is chosen to be much larger than that of the

Algorithm 6 EWPF

```
$\epsilon \leftarrow 0.0001/N$
$\gamma_U \leftarrow 10^{-6}$
$\gamma_N \leftarrow \frac{2^{N_x/2}\epsilon\gamma_U^{N_x}}{\pi^{N_x/2}(1-\epsilon)}$
$N_k \leftarrow N\rho$
for $j = 1, ..., N$ do
    $\mathbf{d}_j \leftarrow \mathbf{y} - \mathbf{H}\left(f(\mathbf{x}_j^{m-1})\right)$
    $c_j \leftarrow -\log \mathbf{w}^{m-1} + 0.5\mathbf{d}_j^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j$
end for
$(\hat{\mathbf{c}}, \mathbf{idx}) \leftarrow sort(\mathbf{c})$
$C_{max} \leftarrow \hat{\mathbf{c}}(N_k)$
for $j = 1, ..., N_k$ do
    $i \leftarrow \mathbf{idx}(j)$
    $a_i \leftarrow \frac{1}{2}\mathbf{d}_i^T\mathbf{R}^{-1}\mathbf{HQH}^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j$
    $b_i \leftarrow \frac{1}{2}\mathbf{d}_i^T\mathbf{R}^{-1}\mathbf{d}_i - C_{max} - \log \mathbf{w}^{m-1}$
    $\alpha_i \leftarrow 1 + \sqrt{1 - b_i/a_i}$
    $\boldsymbol{\beta}_i \sim (1-\epsilon)\mathbf{Q}^{1/2}\mathbf{U}\left(-\gamma_U\mathbf{I}, +\gamma_U\mathbf{I}\right) + \epsilon N\left(\gamma_N^2\mathbf{Q}\right)$
    $\mathbf{x}_j^a \leftarrow f\left(\mathbf{x}_i^{m-1}\right) + \alpha_i\mathbf{QH}^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j + \boldsymbol{\beta}_i$
    if $\boldsymbol{\beta}_i$ was from uniform distribution then
        $\tilde{c}_j \leftarrow -\log \mathbf{w}_i^{m-1} + (\alpha_i^2 - 2\alpha_i)a_i + \frac{1}{2}\mathbf{d}_i^T\mathbf{R}^{-1}\mathbf{d}_i$
    else
        $v_1 \leftarrow -\log \mathbf{w}_i^{m-1} + (\alpha_i^2 - 2\alpha_i)a_i$
        $v_2 \leftarrow v_1 + \frac{1}{2}\mathbf{d}_i^T\mathbf{R}^{-1}\mathbf{d}_i\left(2^{-N_x/2}\right)\left(\pi^{N_x/2}\right)$
        $v_3 \leftarrow v_2\gamma_N\gamma_U^{-N_x}\left(\frac{1-\epsilon}{\epsilon}\right)$
        $\tilde{c}_j \leftarrow v_3 \exp\left(0.5\boldsymbol{\beta}_i^2\right)$
    end if
end for
$\mathbf{w} = \exp(-\tilde{\mathbf{c}})$
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
Resample to have full ensemble, $\mathbf{X}^a$, of $N$ particles from $N_k$ particles $\mathbf{x}^a$.
```

Gaussian. This means that drawing from the Gaussian and also drawing from its tails becomes highly unlikely. In practice, since we always work with small ensemble sizes the chance of filter degeneracy by drawing from the Gaussian, and then drawing from the tail of the Gaussian is indeed highly unlikely.

Finally, the full weights for the new particles are calculated and the whole ensemble is resampled, including those particles that were unable to reach the target weight. Because of the target-weight construction the weights of the particles are very similar, and filter degeneracy is avoided. This filter has been used in a reduced-gravity ocean model by Ades and van Leeuwen (2015b), and in the same system studied for the gravity-wave production by the scheme in Ades and van Leeuwen (2015a). It has also been applied in a climate model by Browne and van Leeuwen (2015).

To analyse the scheme further, we can again look at the variance of the weights. For this it is important to note that this scheme does not see the weight of a particle as a function of the state $\mathbf{X}$ and particle index I, but rather the state as function of the weight W and index I, so $\mathbf{X}(W, I)$. Specifically, $W|I$ has values in two ranges. For the particles with $I = i$ that can reach the target weight we find $w|I = w_{target} + \epsilon_i$ in which ϵ_i is a small perturbation from the target weight due to the small stochastic move discussed above. For those particles that cannot reach the target weight their weights are very close to zero. So we find:

$$E_I[W] \approx \rho(w_{target} + \bar{\epsilon}) + (1-\rho)0 = \rho(w_{target} + \bar{\epsilon}) \tag{56}$$

in which $\bar{\epsilon} = E_I[\epsilon]$. If H is linear and the errors in the observations and the model equations are Gaussian we find $\bar{\epsilon} = 0$, but if any of these three conditions does not hold this is not necessarily so. However, we do know that by construction $|\bar{\epsilon}| << 1$. Since the sum of the weights should be equal to 1, we find that $w_{target} \approx 1/(N\rho)$, and hence $E_I[W] = 1/N$, as expected. Furthermore

$$\begin{aligned} var_I(W) &= \rho\sum_{i=1}^{\rho N}(w_{target} + \epsilon_i)^2 - (\rho w_{target})^2 \\ &\approx \frac{1}{N^2}\frac{1-\rho}{\rho} \end{aligned} \tag{57}$$

This expression shows that the variance in the weights ranges between 0 for $\rho = 1$, so when all particles are kept, to $(N-1)/N^2 \approx 1/N$ for $\rho = 1/N$, so when one particle is kept. We can compare this with the optimal proposal when the number of independent observations is large. In that case one particle will have a weight very close to one, and the rest will have weights very close to zero. The variance in the weights is then $(N-1)/N^2 \approx 1/N$, indeed equal to the $\rho = 1/N$ case in the EWPF scheme, as expected. The EWPF can, however, reduce that variance, even to zero, depending on the choice of the tuning parameter ρ.

When this tuning parameter is chosen close to one, the target weight will be low, and hence particles will be moved further away from the mode of the optimal proposal density. In practise this means that the particles are pushed further away from each other, leading to a wider posterior pdf. A small value for the fraction will have the opposite effect. Since we do not know a-priori what the width of the posterior should be, this is a clear drawback of this method. We will come back to this later.

2.7. *The Implicit Equal-Weights Particle Filter*

In the Implicit Equal Weights Particle Filter (IEWPF) we set the target weight equal to the minimum of the optimal proposal weights for all particles. Then, the position of each particle is set to the mode of the optimal proposal density plus a scaled random perturbation. The scale factor is chosen such that the weight of each particle is equal to the target weight. Note that in the standard setting no resampling is needed, but see Zhu *et al.* (2016) for other possibilities.

The implicit part of the scheme follows from drawing samples implicitly from a standard Gaussian distributed proposal density $q(\boldsymbol{\xi})$ instead of the original $q(\mathbf{x}^n|\mathbf{x}^{n-1},\mathbf{y}^n)$, following the same procedure as in the Implicit Particle Filter. We define a relation

$$\mathbf{x}_i^n = \mathbf{x}_i^a + \alpha_i^{1/2}\mathbf{P}^{1/2}\boldsymbol{\xi}_i^n \tag{58}$$

where $\mathbf{x}_i^a$ is the mode of $p(\mathbf{x}^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)$, $\mathbf{P}$ is a measure of the width of that pdf, $\boldsymbol{\xi}_i^n \in \Re^{N_x}$ is a standard Gaussian-distributed random vector, and α_i is a scalar.

The IEWPF scheme is different from the Implicit Particle Filter in that it chooses the α_i such that all particles get the same weight w_{target}, so the scalar α_i is determined for each particle from:

$$w_i(\alpha_i) = \frac{p(\mathbf{x}_i^n|\mathbf{x}_i^{n-1},\mathbf{y}^n)p(\mathbf{y}^n|\mathbf{x}_i^{n-1})}{Np(\mathbf{y}^n)q(\mathbf{x}^n|i,\mathbf{x}_{1:N}^{n-1},\mathbf{y}^n)} = w_{target} \tag{59}$$

This target weight is equal to the lowest weight over all particles in an optimal proposal. This ensures that the filter is not degenerate in systems with arbitrary dimensions and an arbitrary number of independent observations. The resulting equation for each α_i is nonlinear and complex because it will contain the Jacobian of the transformation from $\boldsymbol{\xi}^n$ to $\mathbf{x}^n$, similar to the Implicit Particle Filter. The Jacobian will contain the derivative of α_i to $\boldsymbol{\xi}_i$, which is the main source of the complexity in this scheme. Algorithm 7 depicts the scheme for the case of a linear observation operator. A nonlinear observation operator will lead to more complicated equations for the α's.

The scheme is similar to the optimal proposal density using the Implicit Particle Filter by first determining the mode of the proposal and then adding a random vector. The difference is that in the IEWPF the size of the vector is determined such that the each particle reaches the target weight. It turns out that this construction excludes part of state space for all but one particle. For each particle the excluded part is different, so the ensemble samples the whole space, but the individual particles do not. Details of the method can be found in Zhu *et al.* (2016).

Algorithm 7 IEWPF

```
for $j = 1, ..., N$ do
    $\mathbf{d}_j \leftarrow \mathbf{y} - \mathbf{H}\left(f(\mathbf{x}_j^{m-1})\right)$
    $c_j \leftarrow -log\mathbf{w}^{m-1} + 0.5\mathbf{d}_j^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j$
end for
$c_{target} \leftarrow \min(\mathbf{c})$
$\mathbf{P} \leftarrow (\mathbf{Q}^{-1} + \mathbf{H}^T\mathbf{R}^{-1}\mathbf{H})^{-1}$
$\boldsymbol{\xi}_i \sim N(0, \mathbf{P})$
for $j = 1, ..., N$ do
    $\mathbf{x}_j^a \leftarrow f\left(\mathbf{x}_j^{m-1}\right) + \mathbf{QH}^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j$
    $\gamma_j \leftarrow \boldsymbol{\xi}_j^T\boldsymbol{\xi}_j$
    $a_j \leftarrow \mathbf{d}_j^T\left(\mathbf{HQH}^T + \mathbf{R}\right)^{-1}\mathbf{d}_j + \log\mathbf{w}^{m-1} + c_{target}$
    Solve $(\alpha_j - 1)\gamma_j - N_x\log\alpha_j + a_j = 0$ for $\alpha_j$
    $\mathbf{x}_j^n \leftarrow \mathbf{x}_j^a + \alpha_j\boldsymbol{\xi}_j$
end for
```

Analysing the scheme in more detail, the proposal density used in this scheme is of one dimension lower than that of the state itself. The *direction* of the random vector in state space is determined by the proposal density, but the *size* of the random vector is then determined deterministically, dependent on that direction. So the proposal density misses one degree of freedom for all but one particle, the particle with the lowest weight that has $\alpha_i = 1$. Although missing one degree of freedom in a very high dimensional system might seem acceptable it does lead to a bias. Figure 5 shows how the implicit equal-weights particle filter works.

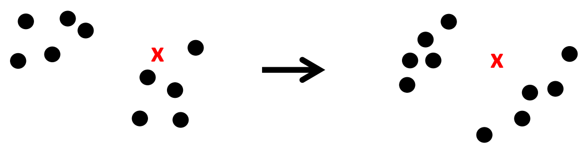

Figure 5. The implicit equal-weights particle filter. Left: the prior particles at time $n-1$(dots), with one observation, denoted with the red cross. Right: the posterior particles. Note that the weights are equal, but some particles have moved away from the observations to ensure equal weights.

2.8. *Discussion*

We first note that the optimal proposal is only optimal in a very limited sense, as has been known a long time with the invention of the auxiliary particle filter. We have seen that it is not difficult to generate particle filters that even have zero variance in the weights. In the optimal proposal setting one forces $Prob(I = i) = 1/N$, while the simple choice $Prob(I = i) \propto p(y^n|x_i^{n-1})$ leads to an equal-weight particle filter. Furthermore, schemes have been introduced that consider the state as function of the state at the previous time and the weight the state at the current time should obtain, so instead of working with $W(\mathbf{X}, I)$ we choose $\mathbf{X}(W, I)$, which opens up a whole new range of efficient particle filters in high dimensional systems.

The EWPF and the IEWPF are by construction particle filters that are not degenerate in high-dimensional systems and that do not rely on localisation. However, it is easy to see that both filters are biased, or inconsistent. In the limit of an infinite number of particles the target-weight constructions will prevent the schemes to converge to the full posterior pdf. The schemes are only of interest when the ensemble size is limited. As long as the bias from the target-weight construction is smaller than the Monte-Carlo error this bias is of no direct consequence. It will be clear that the number of possible methods that have this property is huge, and much more research is needed to explore the best possibilities.

3. Transportation Particle Filters

In resampling particle filters the prior particles are first weighted to represent the posterior and then transformed to unweighted particles simply by duplicating high-weight particles and abandoning low-weight particles. In transformation particle filters one tries to find a transformation that moves particles from the prior to particles of the posterior in a deterministic manner. A related approach, which uses random transformation steps, is based on tempering the likelihood, which we also discuss in this section.

3.1. *One-step transportation*

In one-step transportation one tries to transform samples from the prior into samples from the posterior in one transformation step. An example is the Ensemble Transform Particle Filter (ETPF, Reich 2013), in which the unweighted particles are linear combinations of the weighted particles, so one writes:

$$\mathbf{X}^a = \mathbf{X}^f \mathbf{D} \tag{60}$$

in which the matrix $\mathbf{X}^f = (\mathbf{x}_1^f, \cdots, \mathbf{x}_N^f)$ and similar for $\mathbf{X}^a$, and in which $\mathbf{D}$ is a transformation matrix. The only conditions on $\mathbf{D}$ are that $d_{ij} \geq 0$, $\sum_i d_{ij} = 1$ and $\sum_j d_{ij} = w_i N$. These three conditions leave a lot of freedom for all N^2 elements of D, and a useful way to determine them is to ensure minimal overall movement in state space of the particles from prior to posterior. This leads to an optimal transportation problem and is typically solved by minimizing a cost function that penalises movement of particles.

We can see immediately that this method will not work when the weights are degenerate as the solution will be degenerate and all particles have no other choice than move to the prior particle with weight (close to) one. However, the strength of this filter is that it allows for localisation in a very natural way by making the weights, and hence the matrix $\mathbf{D}$, space dependent. The method will be discussed in more detail in Section 4 on localisation. Here we provide the basic algorithm in Algorithm 8.

Algorithm 8 ETPF

```
$w_i = p(\mathbf{y}|\mathbf{x}_i^f)$
$J(T) \leftarrow \sum_{i,j}^N t_{ij} ||\mathbf{x}_i^f - \mathbf{x}_j^f||^2$
Solve $\min_T J(T)$ with $t_{ij} \geq 0$ , $\sum_i^N t_{ij} = \frac{1}{N}$ and $\sum_j^N = w_i$
$\mathbf{x}_j^a \leftarrow N \sum_i \mathbf{x}_i^f t_{ij}^*$
```

The ETPF provides a direct map from prior to posterior particles without explicitly constructing a transformation map. An alternative approach has been suggested in Moselhy and Marzouk (2012), where an approximate transportation map $\tilde{\mathbf{T}}$ is constructed such that $\tilde{\mathbf{T}}$ belongs to certain family of maps and $\tilde{\mathbf{T}}$ is chosen such that the Kullbeck-Leibler divergence between the pdf generated by $\tilde{\mathbf{T}}$ and the posterior pdf is minimized. See Spantini *et al.* (2017) for an efficient implementation in the context of filtering and smoothing for low-dimensional systems.

3.2. *Tempering of the likelihood*

Instead of trying to transform the particles from the prior to particles from the posterior in one step one can also make this a smoother transition. In tempering (Neal (1996), see also DelMoral *et al.* (2006) and Beskos *et al.* (2014)) one factorises the likelihood as follows:

$$p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y}|\mathbf{x})^{\gamma_1}...p(\mathbf{y}|\mathbf{x})^{\gamma_m} \tag{61}$$

with $0 < \gamma_i < 1$ and ensuring that the sum of the γ's is equal to 1. Then the weighting of the particle filter is first done with the first factor, so

$$p_1(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{y}|\mathbf{x})^{\gamma_1}}{p(\mathbf{y})^{\gamma_1}} p(\mathbf{x}) \tag{62}$$

The reason for this is that the likelihood is much less peaked, and hence the degeneracy can be avoided when γ_1 is small enough. Figure 6 illustrates the basic idea.

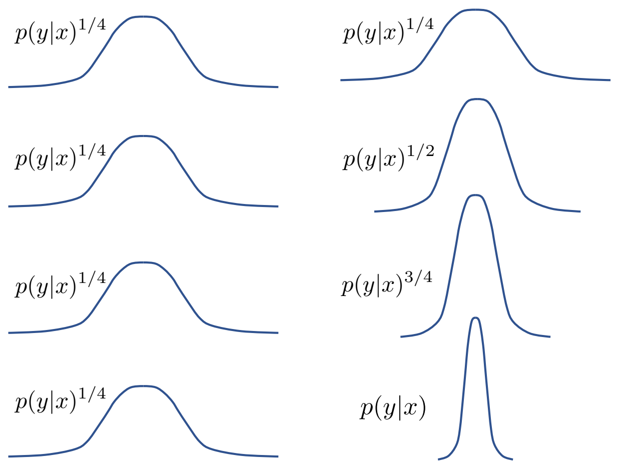

Figure 6. Tempering. The left hand side shows the tempered likelihood functions used in every iteration of the tempering scheme, so every particle filter update. We have chosen $\gamma_i = 1/4$ in this example. The right hand side illustrates how the full likelihood is build up during the tempering process.

The particles are resampled, and now the weighting is performed using the second factor, followed by resampling, etc. In this way the scheme slowly moves all particles towards the high-probability regions of the posterior. Of course, after resampling several particles will be identical, so one needs to jitter the particles, so perturb them slightly, to regain diversity.

This jittering should be a move of the particles that preserves the posterior pdf. It could be implemented as a Markov-Chain Monte-Carlo method with the posterior as the target density, e.g. exploring resample-move strategies, see e.g. Doucet *et al.* (2001). A problem is, however, that in sequential filtering we only have a representation of the posterior density in terms of the present particles, and this representation is very poor due to the small number of particles. Possible avenues are to fit a pdf of a certain shape to the present particles, e.g. a Gaussian mixture model, and use that as target density.

A problem in the geosciences is that this posterior fit needs to preserve the delicate balances between the model variables that are present in each particle, and an extra complication is that these balances can even be nonlinear. Also the transition kernel of the Markov Chain should somehow preserve these balances. An example of its use in the geosciences is the Multiple Data Assimilations (MDA) method of Emerick and Reynolds (2013), in which the intermediate pdf's are assumed to be Gaussian. See also Evensen (2018) for a comparison of this method to other iterative implementations of the Ensemble Kalman Filter/Smoother.

If, however, one allows for model error in the model equations, the following scheme proposed by Beskos *et al.* (2014) does not have this problem. In that case the prior at observation time can be written as (see equation 9):

$$p(\mathbf{x}^n) \approx \frac{1}{N} \sum_{i=1}^{N} p(\mathbf{x}^n | \mathbf{x}_i^{n-1}) \tag{63}$$

in which we assume equal-weight particles at time $n-1$ for ease of presentation. In this case the MCMC method that has the posterior as invariant density is easy to find as the transition densities defined above, followed by an accept/reject step.

When several model time steps are performed between observation times one can also perform tempering in the time domain, as explored in van Leeuwen (2003) and van Leeuwen (2009) in the Guided Particle Filter. The idea is to assimilate the observations ahead of time, with using as likelihood $p(\mathbf{y}^*|\mathbf{x}^m)^\gamma$), in which $\mathbf{y}^*$ is taken equal to the value $\mathbf{y}^n$, and $\gamma << 1$. Here $m < n$ is the present time of the model. This is then followed by a resampling step. The procedure can be followed over several time instances during the forward integration of the particles, increasing γ_i each time. At the observation time $\gamma = 1$ is used. This will force the particles towards the observations and does not need extra jittering because each particle will see a different model noise realisation $\boldsymbol{\beta}$ in the model integration after the resampling steps.

Of course one has to compensate for the fact that the transition density has been changed, and the way to do that is to realise that we have used importance sampling. Instead of sampling from $p(\mathbf{x}^m|\mathbf{x}_i^{m-1})$, we sample from a pdf $q(\mathbf{x}^m|\mathbf{x}_i^{m-1},\mathbf{y}^n) \propto p(\mathbf{x}^m|\mathbf{x}_i^{m-1})p(\mathbf{y}^n|\mathbf{x}^m)^\gamma$, in which $\mathbf{y}^*$ is equal to $\mathbf{y}^n$ taken at time m, and with larger observation uncertainty related to γ. This means that we have to compensate for the weights created by this sampling, so we need to introduce particle weights $w_i^m = p(\mathbf{x}_i^m|\mathbf{x}_i^{m-1})/q(\mathbf{x}_i^m|\mathbf{x}_i^{m-1},\mathbf{y}^*) \propto 1/p(\mathbf{y}^*|\mathbf{x}_i^m)^\gamma$ at each model time step we use this scheme.

The scheme generates extra weights during the model integration, but corrects for them at each new time when we resample, ensuring much better positioned particles at the actual observation time n. It has been used in a reduced-gravity primitive equation model in van Leeuwen (2003), but not in high-dimensional settings.

3.3. *Particle flow filters*

There is a recent surge in methods that dynamically move the particles in state space from equal-weight particles representing the prior, $p(\mathbf{x})$, to equal-weight particles representing the posterior, $p(\mathbf{x}|\mathbf{y})$. In other words, one seeks a differential equation

$$\frac{d}{ds}\mathbf{x} = \mathbf{f}_s(\mathbf{x}) \tag{64}$$

in artificial time $s \geq 0$ with the flow map defining the desired transformation. If the initial conditions of the differential equation (64) are chosen from a pdf $p_0(\mathbf{x})$, then the solutions follow a distribution characterized by the Liouville equation

$$\partial_s p_s = -\nabla_\mathbf{x} \cdot (p_s \mathbf{f}_s)\,. \tag{65}$$

with initial condition $p_0(\mathbf{x}) = p(\mathbf{x})$ and final condition $p_{s_{final}}(\mathbf{x}) = p(\mathbf{x}|\mathbf{y})$.

Two classes of particle flow filters arise. In the first we start from the tempering approach, such that $s_{final} = 1$. We now take the limit of more and more tempering steps by choosing $\gamma_i = 1/n = \Delta s$ with $\lim_{n\to\infty}$, so $\lim_{\gamma_i\to 0}$, or $\lim_{\Delta s\to 0}$, see Daum and Huang (2011, 2013); Reich (2011). This leads to:

$$\begin{aligned} \lim_{\Delta s\to 0} p_{s+\Delta s}(\mathbf{x}) &= p_s(\mathbf{x})\left(\frac{p(\mathbf{y}|\mathbf{x})}{p(\mathbf{y})}\right)^{\Delta s} \\ &= p_s(\mathbf{x})\exp\left[\Delta s\left(\log p(\mathbf{y}|\mathbf{x}) - \log p(\mathbf{y})\right)\right] \\ &\approx p_s(\mathbf{x})\left[1 - \Delta s \log p(\mathbf{y}|\mathbf{x}) - \Delta s \log p(\mathbf{y})\right] \end{aligned} \tag{66}$$

Hence we find:

$$\partial_s p_s(\mathbf{x}) = -\nabla_\mathbf{x} \cdot (p_s \mathbf{f}_s) = p_s(\mathbf{x})(\log p(\mathbf{y}|\mathbf{x}) - c_s) \tag{67}$$

with $c_s = \int p_s(\mathbf{x}) \log p(\mathbf{y}|\mathbf{x}) d\mathbf{x}$. Explicit expression for f_s are available for certain pdfs such as Gaussians and Gaussian mixtures (Reich 2012). These particle flow filters can be viewed as a continuous limit of the tempering methods described in the previous subsection, avoiding the need for resampling and jittering. Note that the elliptic partial differential equation (67) does not determine f_s uniquely. Optimal choices in the sense of minimizing the $L_2(p_s)$–norm of f_s lead to the theory of optimal transportation, see Villani (2008) and Reich and Cotter (2015).

Figure 7 shows the basic idea behind particle flow filters.

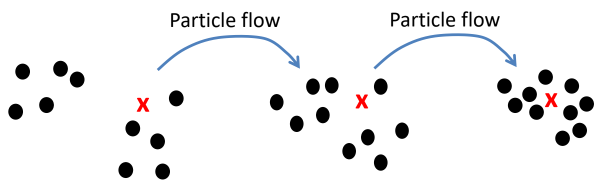

Figure 7. A typical particle flow filter. Left: the prior particles (dots), with one observation, denoted with the red cross. Middle: the particles have moved over several artificial time steps towards the posterior. Note that the weights do not change. Right: the posterior particles after convergence of the filter, sampling the posterior directly.

Alternatively, one can explore ideas from Markov-Chain Monte Carlo (MCMC). One MCMC method that generates samples from the posterior is the Langevin Monte-Carlo sampling, in which a sequence of samples is generated by

$$x^{j+1} = x^j - \Delta s \nabla_\mathbf{x} \log p(\mathbf{x}|\mathbf{y}) + \sqrt{2\Delta s}\beta^j \tag{68}$$

in which β^j a random forcing term drawn from $N(0,\mathbf{I})$. One can show that in the limit of $j \to \infty$ these samples will be samples from the posterior. The corresponding Fokker-Planck equation for this stochastic PDE reads (see, for example, Reich and Cotter

(2015)):

$$
\begin{aligned}
\partial_s p_s &= \nabla_{\mathbf{x}} \cdot (p_s \nabla_{\mathbf{x}}(-\log p(\mathbf{x}|\mathbf{y}))) + \nabla_{\mathbf{x}} \cdot \nabla_{\mathbf{x}} p_s \\
&= -\nabla_{\mathbf{x}} \cdot (p_s \{\nabla_{\mathbf{x}}. \log p(\mathbf{x}|\mathbf{y}) - \nabla_{\mathbf{x}} \log p_s\})
\end{aligned}
$$

This equation corresponds to the deterministic PDE (64) in which $\mathbf{f}_s(\mathbf{x})$ is given by:

$$
\mathbf{f}_s(\mathbf{x}) := \nabla_{\mathbf{x}} \log p(\mathbf{x}|\mathbf{y}) - \nabla_{\mathbf{x}} \log p_s(\mathbf{x}) = -\nabla_{\mathbf{x}} \log \frac{p_s(\mathbf{x})}{p(\mathbf{x}|\mathbf{y})} \tag{69}
$$

Many other choices are possible that use

$$
\lim_{s\to\infty} p_s = p(\mathbf{x}|\mathbf{y}) \tag{70}
$$

in (65). An alternative approach, called Stein variational descent, has recently been proposed by Liu and Wang (2016). Stein variational descent can be viewed as a numerical approximation to a particle flow (64) with vector field

$$
\mathbf{f}_s(\mathbf{x}) := p_s\left(\nabla_{\mathbf{x}} \log p(\mathbf{x}|by) - \nabla_{\mathbf{x}} \log p_s(\mathbf{x})\right) \tag{71}
$$

(Lu *et al.* 2018). We come back to this method below.

In general, to use any of these methods we need to be able to evaluate $p_s(\mathbf{x}_i)$, which is typically unknown as we only know the particle representation of $p_s(\mathbf{x})$. One way to solve this issue is to explore kernel embedding. A numerical implementation of the two formulations (69) and (71) can be based on a reproducing-kernel Hilbert space (RKHS) $\mathcal{F}$ with reproducing kernel $K(.,.)$, typically taken as a Gaussian. In the sequel, we will therefore assume that the kernel is symmetric $K(\mathbf{x},\mathbf{z}) = K(\mathbf{z},\mathbf{x})$. The inner product $\langle g, f\rangle_{\mathcal{F}}$ in $\mathcal{F}$ satisfies the reproducing property

$$
g(\mathbf{x}) = \langle K(\mathbf{x},\cdot), g\rangle_{\mathcal{F}} \,. \tag{72}
$$

A computational approximation to (69) can now be obtained as follows (Russo 1990; Degond and Mustieles 1990). One approximates the pdf p_s by

$$
p_s(\mathbf{x}) = \frac{1}{N}\sum_{j=1}^{N} K(\mathbf{x}_j,\mathbf{x}) \,, \tag{73}
$$

the vector field $\mathbf{f}_s$ by

$$
\mathbf{f}_s(\mathbf{x}) = \frac{\sum_{j=1}^{N} K(\mathbf{x}_j,\mathbf{x})\mathbf{u}_s^j}{p_s(\mathbf{x})} \,, \tag{74}
$$

and the N particles $\mathbf{x}_j$ move under the differential equations

$$
\frac{d}{ds}\mathbf{x}_j = \mathbf{u}_s^j \,. \tag{75}
$$

Since the drift term (69) gives rise to a gradient flow in the space of pdfs with respect to the Kullback–Leibler divergence $\mathrm{KL} = \mathrm{KL}(p_s||p(\cdot|\mathbf{y}))$ between p_s and the posterior pdf (Reich and Cotter 2015), it is natural to introduce the following particle approximation of the Kullback–Leibler divergence:

$$
\mathcal{V}(\{\mathbf{x}_l\}) := \left\langle p_s, \log\frac{p_s}{p(\cdot|\mathbf{y})}\right\rangle_{\mathcal{F}} \,. \tag{76}
$$

in the RKHS $\mathcal{F}$ and to set

$$
\mathbf{u}_s^j := -N\nabla_{\mathbf{x}_j}\mathcal{V}(\{\mathbf{x}_l\}) \tag{77}
$$

in (75), which leads to a gradient flow in the particles $\{\mathbf{x}_l\}$ minimising $\mathcal{V}$. Details on the numerical implementation of this approach can be found in Pathiraja and Reich (2019).

The above formulation restricts the pdf p_s, and hence the prior and the posterior, to be of the form (73). Alternatively, one can embed the vector field of the flow in an appropriate reproducing kernel Hilbert space and not the density itself. With that we can derive a practical implementation of the Stein variational formulation (71) as follows. First, note that the change in KL due to the flow field $\mathbf{f}_s$ can easily be found as:

$$
\begin{aligned}
dKL &= \lim_{\epsilon\to 0}\frac{\mathrm{KL}(p_{s+\epsilon}) - \mathrm{KL}(p_s)}{\epsilon} \\
&= -\int p_s(\mathbf{x})\left[\mathbf{f}_s(\mathbf{x})^T\nabla_{\mathbf{x}}\log p(\mathbf{x}|\mathbf{y}) + \nabla_{\mathbf{x}}\cdot\mathbf{f}_s(\mathbf{x})\right] d\mathbf{x}. \\
&= \langle\nabla\mathrm{KL}, \mathbf{f}_s\rangle_{\mathcal{F}} \,.
\end{aligned} \tag{78}
$$

where $\nabla\mathrm{KL}$ is the gradient of KL, the maximal functional derivative of KL at every state vector $\mathbf{x}$ in the RKHS. Note that $\mathcal{F}$ here is different from the Hilbert space used earlier. Maximising this change in KL as function of the flow field $\mathbf{f}_s$ is not trivial in general. However, with the reproducing kernel property of $\mathbf{f}_s$ we have

$$\mathbf{f}_s(\mathbf{x}) = \langle \mathcal{K}(\cdot, \mathbf{x}), \mathbf{f}_s(\cdot) \rangle \tag{79}$$

in which $\mathcal{K}$ is a vector-valued kernel, typically taken as $\mathcal{K} = \mathbf{I}K$. Using this in (78), the gradient of the KL divergence is found as

$$\nabla \mathrm{KL}(\mathbf{x}) = -\int p_s(\mathbf{z}) \left[K(\mathbf{z}, \mathbf{x}) \nabla_{\mathbf{z}} \log p(\mathbf{z}|\mathbf{y}) + \nabla_{\mathbf{z}} K(\mathbf{z}, \mathbf{x}) \right] d\mathbf{z}\,. \tag{80}$$

The important point is that this gradient is independent from $\mathbf{f}_s$. One now chooses $\mathbf{f}_s$ along this direction, which gives the steepest descent, as

$$\mathbf{f}_s(\mathbf{x}) = -\epsilon \nabla \mathrm{KL}(\mathbf{x}) \tag{81}$$

Finally, one replaces the integral in (80) by its empirical approximation, to obtain

$$\mathbf{f}_s(\mathbf{x}_j) = \epsilon \frac{1}{N} \sum_{l=1}^{N} \left[K(\mathbf{x}_l, \mathbf{x}_j) \nabla_{\mathbf{x}} \log p(\mathbf{x}_l|\mathbf{y}) + \nabla_{\mathbf{x}} K(\mathbf{x}_l, \mathbf{x}_j) \right] \tag{82}$$

for the dynamics (64) of the N particles $\mathbf{x}_j$.

The intuition behind Stein variational descent is that the first term in (82) pulls the particles towards the mode of the posterior, while the second term acts as a repulsive force that allows for particle diversity. Liu and Wang (2016) derived this formulation for a steady-state problem, and Pulido and van Leeuwen (2018) have extended the method to sequential particle filters. The scheme is given in Algorithm 9.

Algorithm 9 Mapping Particle Filter

```
for $j = 1, N$ do
    $\mathbf{x}_j^{k,0} \leftarrow f(\mathbf{x}_j^{k-1}, \beta^k)$
end for
$i = 1$
repeat
    for $j = 1, N$ do
        $\nabla KL(\mathbf{x}) \leftarrow -\frac{1}{N} \sum_{l=1}^{N} \Big[ K(\mathbf{x}_l^{k,i-1}, \mathbf{x}) \nabla \log p(\mathbf{x}_l^{k,i-1}|\mathbf{y})$
            $+ \nabla_x K(\mathbf{x}_l^{k,i-1}, \mathbf{x}) \Big]$
        $\mathbf{x}_j^{k,i} \leftarrow \mathbf{x}_j^{k,i-1} - \epsilon \nabla KL(\mathbf{x}_j^{k,i-1})$
    end for
    $i \leftarrow i + 1$
until Stopping criterion met
```

The free parameter of these methods is the reproducing kernel $K(.,.)$, which needs to be chosen such that the particles sample the posterior and that physical (and potentially other) balances are retained. One also needs to select a proper time stepping scheme, typically chosen as a forward Euler scheme with variable time step ϵ, which can now be viewed as the step length in a gradient descent optimisation algorithm.

3.4. *Discussion*

Viewing particle filters as a transportation problem from equal-weight particles of the prior to equal-weight particles of the posterior has led to an interesting set of filters. None of them have been implemented yet in high-dimensional settings, but some of them are ready to do so. The strong involvement of the machine learning community in problems of this kind also suggests rapid progress here. Finally we mention that the equal-weight particle filters from section 2 can be viewed as one-step transportation filters that explore the proposal density freedom, and in fact transform equal-weight prior particles at time $n-1$ to equal-weight posterior particles at observation time n.

4. Localisation in Particle Filters

Localisation is a standard technique in Ensemble Kalman filtering to increase the rank of the ensemble perturbation matrix, allowing for more observations to be assimilated, and to suppress spurious correlations where real correlations are very small, but ensemble correlations are larger because of sampling noise. Localisation limits the influence of each observation to a localisation area that is much smaller than the full model domain. This idea can easily be incorporated when calculating the particle weights locally, as pioneered by Bengtsson *et al.* (2003), and van Leeuwen (2003), and used in a high-dimensional parameter estimation problem in Vossepoel and van Leeuwen (2006). The difficulty, as we shall see, lies in the resampling step: how does one generate 'smooth' global particles from locally resampled particles. Smooth is not well defined here, but it is related to the particles having realistic physical relations (balances) between the model variables. For example, if geostrophic balance is dominant, the resampling procedure should not generate particles that are completely out of geostrophic balance as that would lead to spurious adjustment processes via spurious gravity waves. Up to now localisation is mainly used in connection with the standard particle filter, while more advanced proposals, apart from the optimal proposal, have not been explored. Farchi and Bocquet (2018) provide an excellent review of localisation in particle filtering, treating a subset of the methods presented here, but including interesting extensions of the methods they describe.

The formal way localisation can be introduced in particle filtering is as follows. Let us denote the state at grid point k as $\mathbf{x}^k$. Hence in contrast to other sections a superscript here denotes not the time index, but the grid point. Note that in geoscience applications each grid point typically has several model variables, so $\mathbf{x}^k$ is a vector in general. Physically it makes sense to assume that the posterior of the state at this grid point depends only on a subset of the observations. Let us denote that subset as $\mathbf{y}^{[k]}$. We can then write:

$$p(\mathbf{x}^k|\mathbf{y}) \approx p(\mathbf{x}^k|\mathbf{y}^{[k]}) \tag{83}$$

In turn, these observations do not depend on the whole state vector but only on part of it, denoted by $\mathbf{x}^{(k)}$:

$$p(\mathbf{y}^{[k]}|\mathbf{x}) = p(\mathbf{y}^{[k]}|\mathbf{x}^{(k)}) \tag{84}$$

Introduce the notation $\mathbf{x}^{(k)\backslash k}$ to denote all those grid points in that part of the state vector excluding grid point k. Then we can rewrite the above as an integral over the joint pdf:

$$p(\mathbf{x}^k|\mathbf{y}^{[k]}) = \int p(\mathbf{x}^{(k)}|\mathbf{y}^{[k]})\; d\mathbf{x}^{(k)\backslash k} \tag{85}$$

Exploring Bayes Theorem we find:

$$\begin{aligned} p(\mathbf{x}^{(k)}|\mathbf{y}^{[k]}) &= \frac{p(\mathbf{y}^{[k]}|\mathbf{x}^{(k)})}{p(\mathbf{y}^{[k]})} p(\mathbf{x}^{(k)}) \\ &\approx \frac{1}{N}\sum_i^N \frac{p(\mathbf{y}^{[k]}|\mathbf{x}_i^{(k)})}{p(\mathbf{y}^{[k]})}\delta(\mathbf{x}^{(k)}-\mathbf{x}_i^{(k)}) \\ &= \sum_i^N w_i^{(k)}\delta(\mathbf{x}^{(k)}-\mathbf{x}_i^{(k)}) \end{aligned} \tag{86}$$

Taken together, this shows that

$$p(\mathbf{x}^k|\mathbf{y}^{[k]}) \approx \sum_i^N w_i^{(k)}\delta(\mathbf{x}^k-\mathbf{x}_i^k) \tag{87}$$

The weights w_i^k thus depend only on the local observations $\mathbf{y}^{[k]}$ and the local prior particles $\mathbf{x}_i^{(k)}$, so that the variance of the weights will be much smaller. Figure 8 illustrates how this local weighting could look for two different particles.

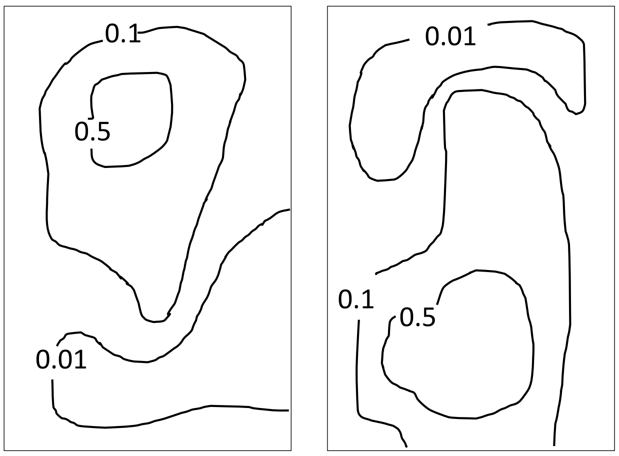

Figure 8. Illustration of a possible local weight distribution in a two-dimensional domain, for two different particles. The particle on the left is close to observations in the central upper part of the domain, leading to high weights there, while the particle on the right is closer to observations in the central lower part of the domain, and hence higher weights there.

The approximation (83) is not unrealistic: a temperature observation in New York is not expected to change our pdf of the temperature in London at the moment of the observation. There will be, of course, an effect at later times, but that is not relevant here. The same assumption underlies the use of localisation in Ensemble Kalman Filters, and in variational methods when the background error covariance is constructed.

However, mathematically it does not follow from the assumption that under the prior the values of the state at grid points separated by more than a certain distance are independent. There can be an indirect flow of information from observations far apart over observations between neigboring grid points. In Ensemble Kalman Filters, the Kalman gain is generally a dense matrix even if $\mathbf{H}\mathbf{P}^b\mathbf{H}^T + \mathbf{R}$ is sparse, because its inverse $(\mathbf{H}\mathbf{P}^b\mathbf{H}^T + \mathbf{R})^{-1}$ can be dense. On the other hand, if $\mathbf{H}\mathbf{P}^b\mathbf{H}^T + \mathbf{R}$ is diagonally dominant, then often its inverse is too.

Repeating the localisation procedure for all grid points, we obtain all marginals of the posterior pdf. However, because the weights $w_i^{(k)}$ change from one grid point to the next, it is non-trivial to obtain a consistent posterior for pairs of state values $(\mathbf{x}^k, \mathbf{x}^\ell)$ (and similarly for triplets etc.). This can easily be seen using Fig. 5: we would like to retain the left particle in the central upper half of the domain, and abandon elsewhere. That would mean that where ever it is abandoned we need to replace it with another particle, perhaps partly with the particle in the right part of the figure. At the boundary between particles a discontinuity will exist, which will lead to unphysical behaviour when this new particle is propagated forward in time.

This means that to obtain global particles that can be forwarded with the model equations one would need to somehow smoothly glue different particles together. This is a major problem and has hampered localisation in particle filtering since the early 2000's. However, recently clever smoothing schemes have been constructed that seem to work well in high-dimensional geophysical applications. We will report on those below.

Another issue is that the localisation area cannot be too large to avoid filter collapse. As a rule of thumb, when there are more than say 10 independent observations inside a local area, the particle filter will still tend to be degenerate for the number of $O(10-1000)$ particles one can typically afford. This means that when the observation density is high the localisation areas have to become unphysically small, or observations have to be discarded. This issue might be solved using tempering techniques as discussed earlier, but is often avoided by artificially enforcing a minimal weight of the particles, or by changing the observations, for instance by projecting them on a lower dimensional space favoured by the prior.

Setting a minimal weight or projecting observations to a lower dimensional space favoured by the prior has as consequence that not all information will be extracted from the observations, as observations that are very different from the existing particles will be largely ignored. This is not directly equivalent to the standard quality control measures used by operational weather forecasting centres, in which observations that are a few standard deviations away from the forecast are ignored. The issue here is that a distance of less then one standard deviation for a few observations can already lead to weight collapse, and artificially setting minimum values for the weights avoids that.

4.1. *Localisation based on resampling*

Several localisation schemes have been proposed and discussed in the review by van Leeuwen (2009) and those will not be repeated here. The most obvious thing to do is to weight and resample locally, and somehow glue the resampled particles together via averaging at the edges between resampled local particles (van Leeuwen 2003). In the following, several schemes in this category are discussed.

4.1.1. *The Localized Particle Filter*

Recently, Penny and Miyoshi (2016) used this idea with more extensive averaging, and their scheme runs as follows. First, for each grid point j the observations close to that grid point are found and the weight of each particle i is calculated based on the likelihood of only those observations:

$$w_{i,j} = \frac{p(\mathbf{y}_j|\mathbf{x}_{i,j})}{\sum_{k=1}^{N} p(\mathbf{y}_j|\mathbf{x}_{k,j})} \tag{88}$$

in which $\mathbf{y}_j$ denotes the set of observations within the localisation area. Note the change of notation from the previous section, related to the explicit use of the particle index in all the following. This is followed by resampling via Stochastic Universal Resampling to provide ensemble members $\mathbf{x}^a_{i,j}$ with $i = 1, ..., N$ for each grid point j.

Farchi and Bocquet (2018) extended this methodology by updating blocks of grid points locally, and introduce a smoothing operator in the weights (similar to Poterjoy (2016)), as:

$$w_{i,j} = \frac{\sum_{k=1}^{N_j} G(d_{j,k}/h)(p(\mathbf{y}_k|\mathbf{x}_{i,k})}{\sum_{m=1}^{N}\sum_{k=1}^{N_j} G(d_{j,k}/h)p(\mathbf{y}_k|\mathbf{x}_{m,k})} \tag{89}$$

in which $G(..)$ is a distance weighting function, e.g. a Gaussian or an approximation of that, $d_{j,k}$ is the distance between grid points j and k, for each observation $\mathbf{y}_k$ at grid point k in the neighbourhood of grid point j. The parameter h is a distance radius, another tuning parameter. This formulation can be used for each grid point j, but also for each block of grid points j. They note that G can also be a Gaussian of a Gaussian, such that it works directly on $-\log p(\mathbf{y}_k|\mathbf{x}_{i,k})$.

As mentioned before, the issue is that two neighbouring grid points can have different sets of particles, and smoothing is needed to ensure that the posterior ensemble consists of smooth particles. This smoothing is performed by Penny and Miyoshi (2016) for each grid point j for each particle i by averaging over the Np neighbouring points within the localisation area around grid point j:

$$x^a_{i,j} = \frac{1}{2}x^a_{i,j} + \frac{1}{2Np}\sum_{k=1}^{N_p} x^a_{i,j_k} \tag{90}$$

in which j_k for $k = 1, ..., N_p$ denotes the grid point index for those points in the localisation area around grid point j. The resampling via Stochastic Universal Resampling is done such that the weights are sorted before the resampling, so that high-weight particles are joined up to reduce spurious gradients.

Farchi and Bocquet (2018) also suggest to smooth this operation, as follows:

$$x^a_{i,j} = \alpha x^a_{i,j} + (1-\alpha)\sum_{k=1}^{N_p} G(d_{j,j_k}/h)x^a_{i,j_k} \tag{91}$$

with α a tuning parameter. Note that choosing $\alpha = 1/2$ and $G(d_{j,j_k}/h) = 1/N_p$ we recover the scheme by Penny and Miyoshi (2016).

While these schemes have been shown to solve the degeneracy problem in intermediate dimensional systems with fixed balances, like the barotropic vorticity model, it is unclear how they will perform in complex systems such as the atmosphere in which fronts can easily be smoothed out, and nonlinear balances broken, see e.g. the discussion in van Leeuwen (2009).

4.1.2. *The Local Particle Filter*

A different scheme that involves a very careful process of ensuring smooth posterior particles and retaining nonlinear relations has recently been proposed by Poterjoy (2016). An important difference with the state-space localisation methods discussed above is that observations are assimilated sequentially to avoid the discontinuity issues of the state-space localisation. This makes the algorithm non-parallel, so slower than the state-space localisation methods, but Farchi and Bocquet (2018) demonstrate that a lower root-mean square error (RMSE) can be achieved.

The scheme proceeds as follows. First, adapted weights are calculated for the first element y_1 of the observation vector, as

$$\tilde{w}_i = \alpha p(y_1|\mathbf{x}_i) + 1 - \alpha \tag{92}$$

These weights are then normalised by their sum $\tilde{W}$. Then the ensemble is resampled according to these normalised weights to form particles $\mathbf{x}_{k_i}$.

The scalar α is an important parameter is this scheme, with $\alpha = 1$ leading to standard weighting, and $\alpha = 0$ leading to all weights being equal to 1 (before normalisation). Its importance lies in the fact that the weights are always larger than $1-\alpha$, so even a value close to 1, say $\alpha = 0.99$, leads to a minimum weight of 0.01 that might seem small, but it means that particles that are more then 1.7 observational standard deviations away from the observations have their weights cut off to a value close to $1-\alpha$. This limits the influence the observation can have on the ensemble. Furthermore, the influence of α does depend on the size of the observational error, which is perhaps not what one would like. It is included to avoid loosing any particle.

Now the following is done for each grid point j. For each member i a weight is calculated as

$$\tilde{\omega}_i = \alpha\rho(1,j,r)p(y_1|\mathbf{x}_i) + 1 - \alpha\rho(1,j,r) \tag{93}$$

in which $\rho(..)$ is the localisation function with localisation radius r. These weights are normalised with their sum over the particles, so a normalised weight ω_i for this grid point is obtained. Note, again, the role played by α. Then the posterior mean for this observation at this grid point is calculated as

$$\bar{\mathbf{x}}_j = \sum_{i=1}^{N} \omega_i \mathbf{x}_{i,j} \tag{94}$$

in which $\mathbf{x}_{i,j}$ is the state at grid point j of particle i. Next a number of scalars are calculated that ensure smooth posterior fields (Poterjoy 2016) as detailed in Algorithm 10.

The final estimate becomes:

$$\mathbf{x}^a_{i,j} = \bar{\mathbf{x}}_j + r_{1j}(\mathbf{x}_{k_i,j} - \bar{\mathbf{x}}_j) + r_{2j}(\mathbf{x}_{i,j} - \bar{\mathbf{x}}_j) \tag{95}$$

where k_i is the index of the i's sampled particle. This procedure is followed for each grid point so that at the end an updated set of particles is obtained that have incorporated the first observation. As a next step the whole process is repeated for the next observation, with the small change that $\tilde{\omega}_i$ is multiplied by $\tilde{\omega}_i$ from the previous observation, until all observations have been assimilated. In this way, the full weight of all observations is accumulated in the algorithm. Now the importance of α comes to full light: without α the ensemble would collapse because the $\tilde{\omega}$'s would be degenerate when observations are accumulated.

The final estimate shows that each particle at grid point j is the posterior mean at that point plus a contribution from the deviation of the posterior resampled particle from that mean and a contribution from the deviation of the prior particle from that mean. So each particle is a mixture of posterior and prior particles, and departures from the prior are suppressed. When $\alpha = 1$, so for a full particle filter, we find for grid points at the observation location, for which $\rho(1, j, r) = 1$, that $c_j = 0$, so $r_{2j} = 0$, and $r_{1j} \approx 1$, so indeed the scheme gives back the full particle filter. The basic elements of the scheme are depicted in Algorithm 10.

Algorithm 10 Local Particle Filter

```
for Each observation $l$ do
    for Each particle $i$ do
        $\tilde{w_i} \leftarrow \alpha p(y_l|x_i) + 1 - \alpha$
    end for
    $\tilde{W} \leftarrow \sum \tilde{w_i}$
    Resample $\mathbf{x}_{k_i}$
    for Each grid point $j$ do
        for Each particle $i$ do
            $\omega_i \leftarrow \alpha\rho(l, j, r)p(y_l|x_i) + 1 - \alpha\rho(l, j, r)$
        end for
        $\overline{x} \leftarrow \sum \omega_i x_{i,j}$
        $\sigma^2 \leftarrow \sum \omega_i (x_{i,j} - \overline{x})^2$
        $c \leftarrow \frac{N(1-\alpha\rho(x_j,\mathbf{y}_l,r))}{\alpha\rho(x_j,\mathbf{y}_l,r)\tilde{W}}$
        $r_1 \leftarrow \sqrt{\frac{\sigma_j^2}{\frac{1}{N-1}\sum_{i=1}^{N}(x_{k_i,j}-\bar{x}+c(x_{i,j}-\bar{x}))^2}}$
        $r_2 \leftarrow cr_1$
        for Each particle $i$ do
            $x_{i,j}^a \leftarrow \bar{x} + r_1(x_{k_i,j} - \bar{x}) + r_2(x_{i,j} - \bar{x})$
        end for
    end for
end for
```

At grid points between observations it can be shown that the particles have the correct first and second order moments, but higher-order moments are not conserved. (Farchi and Bocquet (2018) generate a scheme that is quite similar, but they ensure correct first and second moment by exploring the localised covariances between observed and unobserved grid points directly in a regression step.) To remedy this a probabilistic correction is applied at each grid point, as follows. The prior particles are dressed by Gaussians with width 1 and weighted by the likelihood weights to generate the correct posterior pdf. The posterior particles are dressed in the same way, each with weight $1/N$. Then, the cumulative density functions (cdf's) for the two densities are calculated using a trapezoidal rule integration. A cubic spline is used to find the prior cdf values at each prior particle i, denoted by cdf_i. Then a cubic spline is fitted to the other cdf, and the posterior particle i is found as the inverse of its cdf at value cdf_i. See Poterjoy (2016) for details. The result of this procedure is that higher-order moments are brought back into the ensemble between observation points.

This scheme, although rather complicated, is one of the two local particle filter scheme that has been applied to a high-dimensional geophysical system based on primitive equations in Poterjoy and Anderson (2016). The other is the Localised Adaptive Particle Filter discussed below. (van Leeuwen (2003) applied a local particle filter to a high-dimensional quasi-geostrophic system, but that system is quite robust to sharp gradients as it does not allow for gravity waves.)

4.1.3. *The Localised Adaptive Particle Filter*

The *localized adaptive particle filter* (LAPF) is based on the localized version of the ensemble transform (60) following the LETKF described in Hunt *et al.* (2007), see also Reich (2013), with localization in observation space, and resampling in the spirit of Gaussian Mixture filters (Stordal *et al.* 2011). Localization is carried out around each grid point, and a transform matrix $\mathbf{D}$ is calculated for each localization box. We note that, as for the LETKF, the weights given by (7) depend continuously on the box location and the observations.

In a first step, the observations are projected into the space spanned by the prior particles. As mentioned above, this will reduce the information extracted from the observations, but is perhaps less ad-hoc than setting a lower bound on the weights, as for instance used in the LPF. The LAPF carries out local resampling using universal resampling (see e.g. van Leeuwen (2009)).

In a second step, a careful adaptive sampling is carried out in ensemble space around each of the N temporary particles. This scheme runs as follows:

(a) Resampling is carried out based on a (radial) basis function centered at each particle. A simple case would be a Gaussian mixture, where the covariance of each of the centered Gaussians is taken as a scaled version $c\mathbf{P}$ of the local dynamical ensemble covariance $\mathbf{P}$.

(b) The scaling factor c is individually calculated for each box based on the local observation minus background error statistics. For details we refer to Potthast *et al.* (2019). By this, the LAPF guarantees to obtain a spread of the analysis ensemble which is consistent with the local dynamical observation minus background (o-b) statistics and the observation error covariance **R**. Further standard tools from the LETKF literature to control ensemble spread can be employed if needed.

(c) To obtain sufficient smoothness of the fields in physical space, the LAPF uses N global random draws to generate the resampling vectors around each particle in the space of ensemble coefficients. In combination with the fact that the LAPF draws in each box around each particle only – in a globally uniform way modulated by the ensemble covariance $\mathbf{P}$ and the factor c only –, consistency and balance of the fields is achieved with sufficient precision. The scheme is depicted in Algorithm 11.

Algorithm 11 Local Adaptive Particle Filter

```
for Each grid point $j$, and local grid points $k$ do
    Project local $\mathbf{y}_k$ onto space $\{H(\mathbf{x}^n_{1,k}), ..., H(\mathbf{x}^n_{N,k})\}$
    for $i = 1, .., N$ do
        $w_i \leftarrow p(\mathbf{y}_k|\mathbf{x}_{i,k})$
    end for
    $\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
    Resample
end for
$\mathbf{P} \leftarrow Localized(\mathbf{X}\mathbf{X}^T)$
$c < 1$ (depends on o-b statistics, see text)
for $i = 1, .., N$ do
    $\boldsymbol{\beta} \sim N(0, c\mathbf{P})$
    $\mathbf{x}_i \leftarrow \mathbf{x}_i + \boldsymbol{\beta}$
end for
```

The LAPF is the first particle filter that has been implemented and tested in an operational numerical weather prediction context, and we provide a short description of the procedure. The method has been implemented in the data assimilation system DACE (Data Assimilation Coding Environment) of Deutscher Wetterdienst (DWD) Potthast *et al.* (2019). The DACE environment includes a *Local Ensemble Transform Kalman Filter* (LETKF) based on Hunt *et al.* (2007) both for the global ICON model system and the convection permitting COSMO model system of DWD, see Schraff *et al.* (2016)), both of which are run operationally at DWD[*] and build a basis, framework and reference for the LAPF particle filter implementation.

[*]since January 20, 2016 for the global ICON model with 40km global ensemble resolution including a 20km resolved two-way nest over Europe; and since March 21, 2017 for the COSMO model with 2.8km resolution over central Europe

The ensemble data assimilation system is equipped with a variety of tools to control the spread of the ensemble, such as *multiplicative inflation* and *additive inflation*, *relaxation to prior spread* (RTPS), *relaxation to prior perturbations* (RTPP) and stochastic schemes to add spread to soil moisture and sea surface temperature (SST) when needed (details are described in Schraff *et al.* (2016)).

Tests with the LAPF for the global ICON model with 40 particles of 40km global resolution have been successfully and stably run over a duration of one month. Extensive tests on how many particles form the basis for resampling in each localization box have been carried out, the numbers vary strongly over the globe and all heights of the atmosphere, ranging from 1 to N, with relatively flat distribution. Diagnostics and tuning of the system is under development and discussed in Potthast *et al.* (2019). Results show that the quality of the LAPF does not yet reach the scores of the operational global LETKF-EnVAR system, but the system runs stably and forecast scores are about 10-15% behind the current operational system.

4.2. *The Local Ensemble Transform Particle Filter*

This filter uses a classic sequential importance resampling particle filter from a set of forecast particles $\mathbf{x}^f_i$, which can be obtained employing either the standard or the optimal proposals (or any other) and their associated importance weights w^f_i. The particles are then resampled in a statistically consistent manner, which can be characterized by an $N \times N$ stochastic transition matrix $\mathbf{D}$ with the following properties: (i) all entries d_{ij} of $\mathbf{D}$ are non-negative and

$$\sum_{i=1}^{N} d_{ij} = 1\,, \qquad \frac{1}{N}\sum_{j=1}^{N} d_{ij} = w^f_i\,. \tag{96}$$

Let us denote the set of all such matrices by $\mathcal{D}$. Then any $\mathbf{D} \in \mathcal{D}$ leads to a resampling scheme by randomly drawing an element $j^* \in \{1, \dots, N\}$ according to the probability vector $\mathbf{p}_j = (p_{1j}, \dots, p_{Nj}) \in \mathbb{R}^N$ for each $j = 1, \dots, N$. The jth forecast particle $\mathbf{x}^f_j$ is then replaced by $\mathbf{x}^f_{j^*}$ and the new particles $\mathbf{x}^n_j = x^f_{j^*}$, $j = 1, \dots, N$, provide an equally weighted set of particles from the posterior distribution. Note that multinomial resampling corresponds to the simple choice

$$d_{ij} = w_i^{\mathrm{f}}. \tag{97}$$

The ensemble transform particle filter (ETPF) (Reich 2013; Reich and Cotter 2015) is based on the particular choice $\widehat{\mathbf{D}} \in \mathcal{D}$ that minimizes the expected squared Euclidian distance between forecast particles, i.e.,

$$\widehat{\mathbf{D}} = \arg\min_{\mathbf{D}\in\mathcal{D}} \sum_{i,j=1}^{N} d_{ij} \|\mathbf{x}_i^{\mathrm{f}} - \mathbf{x}_j^{\mathrm{f}}\|^2 . \tag{98}$$

It has been shown under appropriate conditions that the variance of a resampling step based on $\widehat{\mathbf{D}}$ vanishes as $N \to \infty$ (McCann 1995; Reich 2013). This fact is utilized by the ETPF and one defines

$$\mathbf{x}_j^{n} = \sum_{i=1}^{N} \mathbf{x}_i^{\mathrm{f}} \widehat{d}_{ij} \tag{99}$$

even for finite particles numbers. Of course, by its very construction, the ETPF underestimates the posterior covariance. However, there are corrections available that lead to second-order accurate implementations (de Wiljes *et al.* 2017). See Section 5.3 for more details.

Following previously introduced notations, localization can now be implemented into the ETPF as follows. For each grid point k, we extract the values of the forecast particle $\mathbf{x}_i^{\mathrm{f}}$ at that grid point and denote them by $\mathbf{x}_i^k$. Using the observations local to this grid point, we calculate localized importance weights w_i^k for $\mathbf{x}_i^k$. Then (98) gives rise to a localized transformation matrix

$$\widehat{\mathbf{D}}^k = \arg\min_{\mathbf{D}\in\mathcal{D}^k} \sum_{i,j=1}^{N} d_{ij} \|\mathbf{x}_i^{k} - \mathbf{x}_j^{k}\|^2 \tag{100}$$

at grid point k with the set $\mathcal{D}^k$ defined by

$$\mathcal{D}^k = \left\{ \mathbf{D} \in \mathbb{R}_+^{N\times N} : \sum_{i=1}^{N} d_{ij} = 1, \sum_{j=1}^{N} d_{ij} = w_i^k N \right\} . \tag{101}$$

Note that the transport cost (distance) $t_{ij} = \|\mathbf{x}_i^k - \mathbf{x}_j^k\|^2$ can be replaced by any other localized cost function. See Chen and Reich (2015) for more details. The transport problem (100) at each grid point can be computationally expensive. Less expensive approximations, such as the Sinkhorn approximation, and their implementation into the localized ETPF (LETPF) are discussed in de Wiljes *et al.* (2017). Farchi and Bocquet (2018) have extended this algorithm to block weighting, similar to their extension of the Local Particle Filter.

The latter authors also defined a local transform particle filter in state space. This involves a transformation, at each grid point, from prior to posterior particles by a transformation, which essentially becomes an anamorphosis step. The prior and posterior probability densities need to be known as continuous densities, and Farchi and Bocquet (2018) use kernel density estimation with the particles as basis. The interesting suggestion is that since the transformation is deterministic and expected to be smooth over the space coordinates, no specific smoothing is needed after the transformation. We refer to their paper for details on this methodology.

4.3. *Space-Time Particle Filters*

The idea to run a particle filter over the spatial domain was introduced by van Leeuwen (2009), and the first algorithm, the Location Bootstrap Filter, was published by Briggs *et al.* (2013). The Space-Time Particle Filter by Beskos *et al.* (2017) improves on this algorithm by removing the jitter step, as explained below. In the following we assume observations at every grid point, but the algorithms can easily be adapted to other observation networks.

The Location Particle Filter of Briggs *et al.* (2013) runs as follows. The grid points are ordered $1, ..., L$, such that points l and $l+1$ are neighbouring grid points for each $l \in 1, ..., L$. In each grid point l we have a sample $\mathbf{x}_{i,l}$ for $i \in 1, ..., N$, and l denotes the grid point number. We start the spatial particle filter at location $l = 1$ by calculating the weight $p(\mathbf{y}_1|\mathbf{x}_{i,1})$ (where the time index is suppressed) for each prior particle i, and perform resampling using these weights over the whole spatial domain. This means that the resampled particles are now samples of $p(\mathbf{x}^{1:L}|\mathbf{y}^1)$. A small amount of jitter is added to avoid identical particles. The choice of this jitter density is again not clear for geophysical applications, more research is needed on this issue.

Then, the algorithm moves to the next grid point, calculates the weights $p(\mathbf{y}_2|\mathbf{x}_{i,2})$, and resamples the full state particles using this weight, generating samples from $p(\mathbf{x}_{1:L}|\mathbf{y}_1, \mathbf{y}_2)$. Again some jitter is needed to avoid ensemble collapse, and the algorithm moves to the next grid point, until all grid points are treated this way. Algorithm 12 describes the computational steps.

Algorithm 12 Location Particle Filter

```
for Each grid point $j$, and local grid points $k$ do
    for $i = 1, .., N$ do
        $w_i \leftarrow p(\mathbf{y}_k|\mathbf{x}_{i,k})$
    end for
    $\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
    Resample
    Define jitter covariance $\mathbf{S}$
    for $i = 1, .., N$ do
        $\boldsymbol{\beta} \sim N(0, \mathbf{S})$
        $\mathbf{x}_{i,j} \leftarrow \mathbf{x}_{i,j} + \boldsymbol{\beta}$
    end for
end for
```

Note that the algorithm does not suffer from artificial sharp gradients because all resampled particles are global particles, but the algorithm will be very sensitive to the choice of the jitter density used after updating the ensemble in each grid point. Furthermore, when prior and posterior are very different, the algorithm will perform poorly, and Briggs *et al.* (2013) propose a smoother variant that employs copulas for numerical efficiency. We will not discuss that variant here.

Beskos *et al.* (2017) introduce the Space-Time Particle Filter. Instead of using a jitter density to avoid identical particles they exploit the spatial transition density $p(\mathbf{x}_l^n|\mathbf{x}_{l-1}^{n,1}, \mathbf{x}_{1:L}^{n-1})$, in which n is the time index and l the spatial index. (In fact, Beskos *et al.* (2017) allow for a proposal density, but we will explain the algorithm with using the prior spatial pdf as proposal.) So they exploit the pdf of the state at time n and grid point l, $\mathbf{x}_l^n$, conditioned on all previous grid points $\mathbf{x}_{1:l-1}^n$ at the same time n, and conditioned on all grid points at time $n-1$, denoted $\mathbf{x}_{1:L}^{n-1}$. They do this by introducing a set of M local particles j, for each global particle i, with $i \in 1, ..., N$.

For each of the global particles i they run the following algorithm over the whole grid:

1 Starting from location $l = 1$ the M local particle filters grow in dimension when moving over the grid towards the final position L. At the first grid point the prior particles at that grid point are used, weighted with the local likelihood $p(\mathbf{y}_1|\mathbf{x}_1)$ and resampled. Let us call these particles $\hat{\mathbf{x}}_{j,1}$, in which j is the index of the local particle, and 1 is the index of the grid point.

2 The mean $\bar{w}^1$ of the unnormalised weights is calculated.

3 For the next grid point each of these M resampled particles are propagated to that grid point by drawing from $p(\mathbf{x}_2|\hat{\mathbf{x}}_{j,1}, \mathbf{x}_{j,1:L}^{n-1})$. Since each of the M particles is drawn independently they will differ and no jittering is needed.

4 Then the unnormalised weights $p(\mathbf{y}_2|\mathbf{x}_2)$ are calculated, and their mean $\bar{w}^2$, followed by a resampling step.

5 This process is repeated until $l = L$, so until the whole space is covered.

6 Finally, the total weight $w_1 = \prod_{l=1}^L \bar{w}^l$ is calculated, which is the unnormalised weight of the 1st global particle.

Algorithm 13 summarises the scheme.

Algorithm 13 Space-Time Particle Filter

```
for $i = 1, .., N$ do
    for Each grid point $j$, and local grid points $k$ do
        for $m = 1, .., M$ do
            $\mathbf{x}_{m,j}^n \sim p(\mathbf{x}_j^n|\mathbf{x}_{1:l-1}^n, \mathbf{x}_{1:L}^{n-1})$
            $\tilde{w}_m \leftarrow p(\mathbf{y}_k|\mathbf{x}_{i,k})$
        end for
        $\bar{w}_{i,j} \leftarrow \frac{1}{M}\sum_{m=1}^M \tilde{w}_m$
    end for
    $w_i \leftarrow \prod_{j=1}^L \bar{w}_{i,j}$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
Resample
```

This procedure is followed N times for each global particle i independently. These global particles are then resampled according to the weight G_i It is still possible that this filter is degenerate, see Beskos *et al.* (2017) for details and potential solutions.

The importance of this filter lies in the fact that there is a formal proof that it converges to the correct posterior for an increasing number of particles, unlike any of the other algorithms discussed. Furthermore, the authors show that degeneracy can be avoided if the number of particles grows as the square of the dimension of the system, indeed much faster convergence than e.g. the optimal proposal density.

4.4. *Discussion*

Following into the footsteps of Ensemble Kalman Filters, exploring localisation in particle filters is a rapidly growing field. But localisation in particle filters is not trivial as there is no automatic smoothing via smoothed sample covariances as in Ensemble Kalman filters. Most local particle filters impose explicit spatial smoothing, which can affect delicate balances in the system. Worth mentioning in this context is the localisation introduced by Robert and Künsch (2017), who process observations sequentially in their hybrid Ensemble Kalman Filter-Particle Filter approach such that the second-order properties of the particle-filter part remain correct. This method is discussed in the next chapter. The Ensemble Transform Particle Filter and the Localized Adaptive Particle Filter come closest to the Ensemble Kalman Filter by using a linear transportation matrix to transforms the prior ensemble into a posterior ensemble, and this matrix can be made smoothly varying with space. All of these smoothing operations rely on forming linear combinations of particles, so can potentially harm nonlinear balances in the model. Furthermore, it should be noted that the smoothing operation does not necessarily follow Bayes Theorem, so it might result in an extra approximation of the true posterior pdf. When the ensemble size is small this approximation might be negligible compared to the Monte-Carlo noise from the finite ensemble size, however.

The Location Particle Filter and the Space-Time Particle Filter avoid this smoothing and rely on statistical connections between different grid points. The former does this via the prior pdf, defined by the prior particles. When the number of particles is low this pdf is estimated rather poorly. Furthermore, the method needs jittering of the global particles to avoid ensemble collapse after every resampling step after each new observation is assimilated. This jittering pdf can be chosen arbitrarily, for instance a smooth Gaussian, but it does violate Bayes Theorem. As mentioned above, this error might be negligible when the ensemble size is small. The latter method explores the transition density over space and time, leading to consistent estimates of the spatial relations between grid points. Another potential issue of both methods is that if the spatial field is two or higher dimensional, as in geoscience applications, it is unclear how to order the grid points, and potentially large jumps might be created between neighbouring grid points that are treated as far apart by the algorithm. This needs further investigation.

5. Hybrids between Particle Filters and Ensemble Kalman Filters

As mentioned in the previous section, there are two issues with localisation. Firstly, particle filters that employ resampling need to ensure smooth updates in space so that the newly formed global particles do not encounter strong adjustments to physical balances due to artificial gradients from glueing particles together. Present-day localised particle schemes concentrate on this issue.

Secondly, the localisation area cannot contain too many independent observations, and as a rule of thumb 10 independent observations is often too many, to avoid weight collapse. As mentioned, this demand can be in strong contrast with physical considerations of appropriate length scales. This is one of the main reasons to consider hybrids between particle filters and ensemble Kalman filters within a localisation scheme. In the following several recent hybrid methods are presented.

5.1. Adaptive Gaussian Mixture Filter

A bridging formulation allows to smoothly transition between an ensemble Kalman filter and a particle filter analysis update. One such formulation is the adaptive Gaussian mixture filter (Stordal *et al.* 2011).

In a Gaussian mixture filter, the distribution is approximated by a combination of normal distributions centered at the values of the particles. Thus we have

$$p(\mathbf{x}^n) = \sum_{i=1}^{N} w_i N\left(\mathbf{x}_i^f, \hat{\mathbf{P}}^f\right) \tag{102}$$

where $N(\mathbf{x}_i^f, \hat{\mathbf{P}}^f)$ is a Gaussian Kernel with mean $\mathbf{x}_i^n$ and covariance $\hat{\mathbf{P}}^f$. This covariance is initialized from the sample covariance matrix $\mathbf{P}^f$ of the ensemble by multiplying with a so-called bandwidth parameter $0 < h \leq 1$ such that

$$\hat{\mathbf{P}}^f = h^2 \mathbf{P}^f. \tag{103}$$

At the analysis time, the filter computes a two-step update: In the first step we update the ensemble members and the covariance matrix according to the Kalman filter equations given by

$$\mathbf{X}^n = \mathbf{X}^f + \hat{\mathbf{K}}^n \left(\mathbf{y}^n \mathbf{1}^T - \mathbf{H}\mathbf{X}^f\right) \quad (104)$$

$$\hat{\mathbf{K}}^n = \hat{\mathbf{P}}^f \mathbf{H}^T \left(\mathbf{H}\hat{\mathbf{P}}^f \mathbf{H}^T + \mathbf{R}^n\right)^{-1} \quad (105)$$

and

$$\mathbf{P}^n = \left(\mathbf{I} - \hat{\mathbf{K}}^n \mathbf{H}\right) \hat{\mathbf{P}}^f. \quad (106)$$

Note that this is just a short-hand notation for updating each centre fo the prior Gaussians. For computational efficiency the analysis equations in the (adaptive) Gaussian mixture filter (Hoteit *et al.* 2008; Stordal *et al.* 2011) were proposed to use a factorized covariance matrix in the form $\hat{\mathbf{P}}^f = \mathbf{LUL}^T$ as can be obtained from a singular value decomposition of the ensemble perturbation matrix and used, e.g. in the SEIK filter (Pham 2001) and error-subspace transform Kalman filter (ESTKF, Nerger *et al.* 2012). However, the particular form of the Kalman filter update equations is not crucial here.

In the second step we update the weights of the particles according to

$$w_i^n \approx w_i^{n-1} N_{\mathbf{y}^n|\mathbf{x}^f} \left(\mathbf{H}\mathbf{x}_i^f, \mathbf{R}^n\right) \quad (107)$$

in which $\mathbf{R}^n = \mathbf{R} + \mathbf{H}\hat{\mathbf{P}}^f \mathbf{H}^T$, and then normalise these so that the sum of the weights is one.

The bridging is now done by interpolating the analysis weight with a uniform weight N^{-1} as

$$w_i^{(\alpha)} = \alpha w_i + (1-\alpha) N^{-1}, \quad (108)$$

where α is the bridging parameter. We obtain a transition between the ensemble Kalman filter and the particle filter by varying both α and h. For $\alpha = 0$ and $h = 1$ we obtain the uniform weights of the ensemble Kalman filter, while for $\alpha = 1$ and $h = 0$ we obtain the particle filter weights. Stordal *et al.* (2011) proposed to adaptively estimate an optimal value of α by setting $\alpha = N^{-1}\hat{N}_{eff}$ where $\hat{N}_{eff} = (\sum_i w_i^2)^{-1}$ is the effective sample size.

The update formulation of the adaptive Gaussian mixture filter reduces the risk of ensemble degeneracy, but cannot fully avoid it. To this end, we can combine the filter with a resampling step as in other particle filters.

5.2. *Ensemble Kalman Particle Filter*

The Ensemble Kalman Particle Filter of Frei and Künsch (2013) is a hybrid EnKF-PF. It is based on tempering in just two steps, splitting the likelihood into two factors

$$p(\mathbf{x}^n|\mathbf{y}^n) = p(\mathbf{x}^n|\mathbf{y}^n)^\alpha \; p(\mathbf{x}^n|\mathbf{y}^n)^{1-\alpha} \quad (109)$$

with $\alpha \in (0, 1)$. In the first step the Stochastic Ensemble Kalman filter of Burgers *et al.* (1998) is applied, and in the second step a particle filter. When the parameter α is close to 0 the scheme is like a full particle filter, while for α close to 1 it is essentially the ensemble Kalman filter. Figure 9 illustrates the idea.

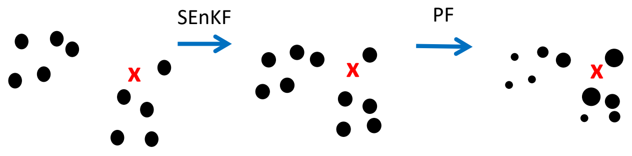

Figure 9. The Ensemble Kalman Particle Filter. First a Stochastic EnKF is performed, followed by a standard Particle Filter.

Two problems with a direct application of the above scheme are identified by Frei and Künsch (2013): the particle filter weights are influenced by the random modelled observations in the Stochastic EnKF (SEnKF), and the resampling step in the particle filter will lead to identical particles. To avoid both, the algorithm is modified as follows. Firstly, assuming a Gaussian likelihood, the SEnKF particles can be written as:

$$\mathbf{x}_i^{SEnKF} = \mathbf{x}_i + \mathbf{K}_\alpha(\mathbf{y} - \mathbf{H}\mathbf{x}_i - \boldsymbol{\epsilon}_i) \quad (110)$$

with $\boldsymbol{\epsilon}_i \sim N(0, \mathbf{R}/\alpha)$ and $\mathbf{K}_\alpha$ is the normal gain, but with $\mathbf{R}$ divided by α. Thus, the particles can be seen as draws from

$$\mathbf{x}_i^{SEnKF} \sim N(\boldsymbol{\nu}_i, \mathbf{P}^{EnKF}) \quad (111)$$

in which

$$\boldsymbol{\nu}_i = \mathbf{x}_i + \mathbf{K}_\alpha(\mathbf{y} - \mathbf{H}\mathbf{x}_i) \quad (112)$$

and

$$\mathbf{P}^{SEnKF} = \frac{1}{\alpha}\mathbf{K}_\alpha \mathbf{R}\mathbf{K}_\alpha^T. \tag{113}$$

Hence the SEnKF posterior can be written as:

$$p(\mathbf{x}|\mathbf{y})^{SEnKF} = \frac{1}{N}\sum_{i=1}^{N} N(\boldsymbol{\nu}_i, \mathbf{P}^{SEnKF}) \tag{114}$$

Instead of performing the standard SEnKF sampling from this density we delay that sampling and perform the multiplication with the second likelihood $p(\mathbf{y}|\mathbf{x})^{1-\alpha}$ analytically. This is easy because the EnKF posterior is a Gaussian mixture and the likelihood is a Gaussian, so the full posterior is a Gaussian mixture too. This leads to a full posterior

$$\sum_{i=1}^{N} \gamma_i N(\boldsymbol{\mu}_i, \mathbf{P}^{PF}) \tag{115}$$

in which

$$\boldsymbol{\mu}_i = \boldsymbol{\nu}_i + \hat{\mathbf{K}}(\mathbf{y} - \mathbf{H}\boldsymbol{\nu}_i) \tag{116}$$

$$\gamma_i = N\left(\mathbf{y} - \mathbf{H}\boldsymbol{\nu}_i, \mathbf{H}\mathbf{P}^{SEnKF}\mathbf{H}^T + \mathbf{R}/(1-\alpha)\right) \tag{117}$$

$$\mathbf{P}^{PF} = (\mathbf{I} - \hat{\mathbf{K}}H)\mathbf{P}^{SEnKF} \tag{118}$$

where

$$\hat{\mathbf{K}} = \mathbf{P}^{SEnKF}\mathbf{H}^T\left(\mathbf{H}\mathbf{P}^{SEnKF}\mathbf{H}^T + \mathbf{R}/(1-\alpha)\right)^{-1} \tag{119}$$

Note that the normalisation constants in γ_i do not have to be calculated as we know that they should fulfil $\sum_i \gamma_i = 1$.

The way to sample the particles now becomes a two step procedure. First draw N samples from the distribution of the mixture coefficients γ_i and then draw from the selected Gaussian mixture components:

$$\mathbf{x}_i^{EKPF} = \boldsymbol{\mu}_{k_i} + \boldsymbol{\xi}_i \tag{120}$$

in which k_i denotes the resampled particle index i and $\boldsymbol{\xi}_i \sim N(0, \mathbf{P}^{PF})$. The variables $\boldsymbol{\xi}_i$ can again be generated in two steps by

$$\boldsymbol{\xi}_i = (\mathbf{I} - \hat{\mathbf{K}}H^T)\mathbf{K}_\alpha\boldsymbol{\epsilon}_{i,1} + \hat{\mathbf{K}}\boldsymbol{\epsilon}_{i,2} \tag{121}$$

where $\boldsymbol{\epsilon}_{1.i}$ and $\boldsymbol{\epsilon}_{i,2}$ are independent draws from $N(0, \mathbf{R}/\alpha)$ and $N(0, \mathbf{R}/(1-\alpha))$, respectively.

The scheme is very closely related to a Gaussian mixture model, as the EnKF step forces the prior for the particle filter to be a Gaussian mixture. The strong point of this scheme is that the width of each Gaussian follows naturally from the stochastic part of the EnKF, while it is ad hoc in standard Gaussian mixture models. Furthermore, while the standard Gaussian mixture model uses the observation covariance matrix $\mathbf{R}$ this filter uses an inflated $\mathbf{H}\mathbf{P}^{SEnKF}\mathbf{H}^T + \mathbf{R}/(1-\alpha)$, which will lead to a better weight distribution. Finally, the starting points of the centres of the prior Gaussians will be closer the observations, suggesting more uniform weights. The pseudocode of the scheme is presented in Algorithm 14.

Algorithm 14 Ensemble Kalman Particle Filter

```
$\mathbf{R}_\alpha \leftarrow \mathbf{R}/\alpha$
$\mathbf{K}_\alpha \leftarrow \mathbf{P}\mathbf{H}^T(\mathbf{H}\mathbf{P}\mathbf{H}^T + \mathbf{R}_\alpha)^{-1}$
$\mathbf{P}^{SEnKF} \leftarrow \frac{1}{\alpha}\mathbf{K}_\alpha\mathbf{R}\mathbf{K}_\alpha^T$
$\hat{\mathbf{K}} \leftarrow \mathbf{P}^{SEnKF}\mathbf{H}^T\left(\mathbf{H}\mathbf{P}^{SEnKF}\mathbf{H}^T + \mathbf{R}/(1-\alpha)\right)^{-1}$
for $i = 1,..,N$ do
    $\boldsymbol{\epsilon}_{i,1} \sim N(0, \mathbf{R}_\alpha)$
    $\boldsymbol{\epsilon}_{i,2} \sim N(0, \mathbf{R}/(1-\alpha))$
    $\boldsymbol{\nu}_i \leftarrow \mathbf{x}_i + \mathbf{K}_\alpha(\mathbf{y} - \mathbf{H}\mathbf{x}_i - \boldsymbol{\epsilon}_i)$
    $\boldsymbol{\mu}_i \leftarrow \boldsymbol{\nu}_i + \hat{\mathbf{K}}(\mathbf{y} - \mathbf{H}\boldsymbol{\nu}_i)$
    $\gamma_i \sim N\left(\mathbf{y} - \mathbf{H}\boldsymbol{\nu}_i, \mathbf{H}\mathbf{P}^{SEnKF}\mathbf{H}^T + \mathbf{R}/(1-\alpha)\right)$
end for
$\boldsymbol{\gamma} \leftarrow \boldsymbol{\gamma}/\boldsymbol{\gamma}^T\mathbf{1}$
for $i = 1,..,N$ do
    $k_i \sim MultiNomial(\boldsymbol{\gamma})$
    $\boldsymbol{\xi}_i \leftarrow (\mathbf{I} - \hat{\mathbf{K}}H^T)\mathbf{K}_\alpha\boldsymbol{\epsilon}_{i,1} + \hat{\mathbf{K}}\boldsymbol{\epsilon}_{i,2}$
    $\mathbf{x}_i^{EKPF} \leftarrow \boldsymbol{\mu}_{k_i} + \boldsymbol{\xi}_i$
end for
Resample
```

In an extension of the scheme, Frei and Künsch (2013) suggest to form a tempering scheme, alternatively using the ensemble Kalman filter and the particle filter. The resampling step of the particle filter is not problematic in this case as the Kalman filter will diversify identical particles in each next iteration. The paper also discusses approximate schemes for non-Gaussian observation errors and nonlinear observation operators.

In Robert *et al.* (2017), a variant of this method has been introduced which is based on the LETKF instead of the stochastic variant and in which the update is in ensemble space:

$$\mathbf{X}^{PI} = \mathbf{X}^f\mathbf{W} \tag{122}$$

where the column sums of $\mathbf{W}$ equal 1. The matrix $\mathbf{W}$ can be split into

$$\mathbf{W} = \mathbf{W}^{\mu}\mathbf{W}^{\alpha} + \mathbf{W}^{\xi} \tag{123}$$

where $\mathbf{W}^{\mu}$ corresponds to computing the centers $\boldsymbol{\mu}_i$, $\mathbf{W}^{\alpha}$ to the resampling and $\mathbf{W}^{\xi}$ to the added noise $\boldsymbol{\xi}_i$. In the transform variant $\mathbf{W}^{\xi}$ is deterministic and chosen such that the sample covariance of $\mathbf{X}^{PI}$ is equals the covariance of the Gaussian mixture (115). It thus belongs also to the class of second-order exact filters discussed in the next section.

Robert *et al.* (2017) apply a localized transform Ensemble Kalman Particle Filter in the KENDA (Kilometer-Scale Ensemble Data Assimilation) system with a setup similar to the one used operationally by MeteoSwiss. This system computes the weight matrices $\mathbf{W}$ only on a coarse grid and then interpolates these matrices to the original grid. Therefore the discontinuities introduced by resampling are smoothed out, but in a way that is possibly optimal for the EnKF and not for the EnKPF. In Robert and Künsch (2017) a different localization method for the EnKPF was developed which proceeds by sequentially assimilating observations y^k, limiting the state components influenced by y^k to a subset. It smoothes out the discontinuities that occur when a resampled particle in the region influenced by y^k is connected to a background particle outside of this region. The smoothing is done in such a way that the second-order properties of the smoothed particle remain correct.

5.3. *Second-order exact filters*

A second-order exact filter ensures that the posterior ensemble mean and ensemble covariance matrix are equal to those obtained from the particle filter weights. Thus, the requirement for the mean of the analysis ensemble is

$$\overline{\mathbf{x}}^n = \frac{1}{N}\sum_{i=1}^{N}\mathbf{x}_i^n = \sum_{i=1}^{N} w_i \mathbf{x}_i^f \tag{124}$$

where the superscript f denotes the forecasted state vector. Likewise, the posterior ensemble covariance matrix is required to fulfil

$$\begin{aligned}\mathbf{P}^a &= \frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{x}_i^n - \overline{\mathbf{x}}^n\right)\left(\mathbf{x}_i^n - \overline{\mathbf{x}}^n\right)^T & (125)\\ &= \sum_{i=1}^{N} w_i \left(\mathbf{x}_i^f - \overline{\mathbf{x}}^n\right)\left(\mathbf{x}_i^f - \overline{\mathbf{x}}^n\right)^T. & (126)\end{aligned}$$

5.3.1. *Merging Particle Filter*

The merging particle filter by Nakano *et al.* (2007) explores the sampling aspect of the resampling step. The method draws a set of q ensembles each of size N from the weighted prior ensemble at the resampling step. Then these sets are merged via a weighted average to obtain a new set of particles that has the correct mean and covariance but is more robust than the standard particle filter. Define $\mathbf{x}_{i,j}$ as ensemble member i in ensemble j. The new merged ensemble members are generated via

$$\mathbf{x}_i^a = \sum_{j=1}^{q}\alpha_j \mathbf{x}_{i,j}. \tag{127}$$

To ensure that the new ensemble has the correct mean and covariance, the coefficients α_j have to be real and need to fulfil the two conditions

$$\sum_{j=1}^{q}\alpha_j = 1; \quad \sum_{j=1}^{q}\alpha_j^2 = 1, \tag{128}$$

When $q > 3$ there is no unique solution for the α's, while for $q = 3$ one finds:

$$\begin{aligned}\alpha_1 &= \frac{3}{4}\\ \alpha_2 &= \frac{\sqrt{13}+1}{8}\\ \alpha_3 &= -\frac{\sqrt{13}-1}{8} & (129)\end{aligned}$$

We can make the weights space-dependent in high-dimensional systems and since the new particles are merged previous particles the resulting global particles are expected to be smooth. The scheme is depicted in Algorithm 15.

5.3.2. *Nonlinear Ensemble Transform Filter NETF*

A simple formulation of a second-order exact filter can be obtained by using Eq. (124) to compute the mean of the posterior

Algorithm 15 Merging Particle Filter

```
for $i = 1, .., N$ do
    $w_i \leftarrow p(\mathbf{y}_k|\mathbf{x}_i)$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
$(\mathbf{X}^a_1, ..., \mathbf{X}^a_q) \leftarrow q$ times resampled prior ensemble
Find $\alpha_i$ such that $\sum_i \alpha_i = 1$ and $\sum_i \alpha_i^2 = 1$
$\mathbf{X}^a \leftarrow \sum \alpha_i \mathbf{X}^a_i$
```

ensemble (Xiong *et al.* 2006; Tödter and Ahrens 2015). For the associated ensemble perturbations, we can derive from Eq. (126) with $\mathbf{w} = (w_1, \ldots, w_N)^T$ and $\mathbf{W} = \text{diag}(\mathbf{w})$ that

$$\mathbf{P}^a = \mathbf{X}^f \left(\mathbf{W} - \mathbf{w}\mathbf{w}^T\right) (\mathbf{X}^f)^T . \tag{130}$$

Posterior ensemble perturbations can now be obtained by factorizing $\mathbf{A} = \mathbf{W} - \mathbf{w}\mathbf{w}^T$, e.g. by a singular value decomposition as $\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$. This leads to $\mathbf{A}^{1/2} = \mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{V}^T$ and posterior perturbations are then given by

$$\mathbf{X}'^n = \sqrt{N}\mathbf{X}^f\mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{V}^T. \tag{131}$$

Finally, the full posterior particles are given by

$$\mathbf{x}^n_i = \mathbf{X}^f \left(\mathbf{w}\mathbf{1}^T + \sqrt{N}\mathbf{V}\mathbf{\Lambda}^{1/2}\mathbf{V}^T\right)_i . \tag{132}$$

The computations of this filter are very similar to those in ensemble square-root Kalman filters like the ETKF (Hunt *et al.* 2007) or ESTKF (Nerger *et al.* 2012). As such, we can can also localize the filter in the same way. The localized NETF has been successfully applied to a high-dimensional geophysical system based on primitive equations in Tödter *et al.* (2016). In addition, the filter can be easily extended to a smoother by applying the filter transform matrix (the term in parenthesis in Eq. 132) to previous analysis times (Kirchgessner *et al.* 2017). The scheme is depicted in Algorithm 16.

Algorithm 16 NETF

```
for $i = 1, .., N$ do
    $w_i \leftarrow p(\mathbf{y}_k|\mathbf{x}_i)$
end for
$\mathbf{w} \leftarrow \mathbf{w}/\mathbf{w}^T\mathbf{1}$
$\mathbf{A} \leftarrow diag(\mathbf{w}) - \mathbf{w}\mathbf{w}^T$
$\mathbf{V}\Lambda\mathbf{V}^T \leftarrow \mathbf{A}$
$\mathbf{T} \leftarrow \sqrt{N}\mathbf{V}\Lambda^{1/2}\mathbf{V}^T$
$\mathbf{T} \leftarrow \mathbf{T} + \mathbf{w}$
$\mathbf{X}^a \leftarrow \mathbf{X}^f\mathbf{T}$
```

5.3.3. *Nonlinear Ensemble Adjustment Filter*

There is also a stochastic variant of the previous algorithm (Lei and Bickel 2011), which is motivated from the Stochastic Ensemble Kalman filter (Burgers *et al.* 1998; Houtekamer and Mitchell 1998). In this filter, we generate a set of perturbed model observations

$$\mathbf{y}_i = \mathbf{H}(\mathbf{x}_i) + \boldsymbol{\epsilon}_i, \quad i = 1, \ldots, N, \tag{133}$$

which represents the observation probability distribution. We now obtain an analysis mean of each particle analogously to Eq. (124) by

$$\overline{\mathbf{x}}^n(\mathbf{y}_k) = \sum_{i=1}^{N} w_i(\mathbf{y}_k)x^f_i \tag{134}$$

where each weight $w_i(\mathbf{y}_k)$ is computed from the likelihood of the perturbed measured ensemble member $\mathbf{H}(\mathbf{x}_i)$. When we now define

$$\hat{\mathbf{P}}^a(\mathbf{y}_k) = \sum_{i=1}^{N} w_i(\mathbf{y}_k) \left(\mathbf{x}^f_i - \overline{\mathbf{x}}^n(\mathbf{y}_k)\right) \left(\mathbf{x}^f_i - \overline{\mathbf{x}}^n(\mathbf{y}_k)\right)^T \tag{135}$$

we obtain the posterior ensemble members as

$$\mathbf{x}^n_k = \overline{\mathbf{x}}^n + (\mathbf{P}^a)^{1/2}\hat{\mathbf{P}}^a(\mathbf{y}_k)^{-1/2}(\mathbf{x}^f_k - \overline{\mathbf{x}}^n(\mathbf{y}_k)) \tag{136}$$

where $\overline{\mathbf{x}}^n$ is given by Eq. (124) and P^a is given by Eq. (126). This update equation only yields the correct first and second moments of the posterior distribution in the limit of a large ensemble.

5.3.4. *Second-order exact ETPF*

Also the ETPF (see Sec. 4.2) can be formulated to be second-order accurate (de Wiljes *et al.* 2017). For this, we approximate

$$\mathbf{A} = \mathbf{W} - \mathbf{w}\mathbf{w}^T \approx \frac{1}{N}\left(\widehat{\mathbf{D}} - \mathbf{w}\mathbf{1}^T\right)\left(\widehat{\mathbf{D}} - \mathbf{w}\mathbf{1}^T\right)^T \tag{137}$$

where the matrix $\widehat{\mathbf{D}}$ is obtained through (98). To ensure the second-order accuracy, we introduce a correction term such that

$$\widetilde{\mathbf{D}} = \widehat{\mathbf{D}} + \mathbf{\Delta} \tag{138}$$

with $\mathbf{\Delta}$ being a symmetric $N \times N$ matrix. Using $\widetilde{\mathbf{D}}$ in Eq. (137) and requiring that the result is equal to $\mathbf{A}$ leads to the condition

$$N(\mathbf{W} - \mathbf{w}\mathbf{w}^T) - (\widehat{\mathbf{D}} - \mathbf{W}\mathbf{1}^T)(\widehat{\mathbf{D}} - \mathbf{W}\mathbf{1}^T)^T \tag{139}$$

$$= (\widehat{\mathbf{D}} - \mathbf{W}\mathbf{1}^T)\mathbf{\Delta} + \mathbf{\Delta}(\widehat{\mathbf{D}} - \mathbf{W}\mathbf{1}^T)^T + \mathbf{\Delta}\mathbf{\Delta}, \tag{140}$$

which is a quadratic equation in $\mathbf{\Delta}$ in the form of a continuous-time algebraic Riccati equation and there are known solution methods for this type of equation (see, e.g., de Wiljes *et al.* 2017). Note that $\widetilde{\mathbf{D}}$ still satisfies (96). However, $\widetilde{d}_{ij} \geq 0$ does not hold anymore, in general.

5.4. *Hybrid LETPF-LETKF*

The hybrid LETPF-LETKF is also based on the simple idea of splitting the likelihood function into two factors at each grid point k, i.e.

$$p(\mathbf{x}^k|\mathbf{y}^{(k)}) = p(\mathbf{x}^k|\mathbf{y}^{(k)})^{1-\alpha}\; p(\mathbf{x}^k|\mathbf{y}^{(k)})^{\alpha} \tag{141}$$

with $\alpha \in (0,1)$, but now the particle filter is employed first, followed by the ensemble Kalman filter. This is similar to tempering in just two steps. When the likelihood is Gaussian the posterior is expected to be more Gaussian than the prior. Hence it makes sense to use a particle filter in the first step, and to try to use an EnKF in the second step of the tempering procedure.

If the likelihood is Gaussian with localized error covariance matrix $\mathbf{R}^k$, then the factorization is equivalent to scaling this matrix by $1/\alpha$ and $1/(1-\alpha)$, respectively. Hence, one can, for example, first apply an LETPF to the forecast particles $\mathbf{x}_i^{\mathrm{f}}$ with inflated covariance matrix $\mathbf{R}^k/\alpha$ in order to obtain new particle values

$$\tilde{\mathbf{x}}_i^k = \sum_{j=1}^{N} d_{ij}^k(\alpha)\mathbf{x}_i^k \tag{142}$$

at each grid point k. One then applies the LETKF to these intermediate particles $\tilde{\mathbf{x}}_i$ with inflated covariance matrix $\mathbf{R}^k/(1-\alpha)$. The choice of α is, of course, crucial. Numerical experiments indicate (Chustagulprom *et al.* 2016) that $\alpha > 0$ can lead to substantial improvements over a purely LETKF-based implementation and that the choice of α can be based on the effective sample size of the associated LETPF. However, more refined selection criteria for the parameter α are needed to make the hybrid LETPF-LETKF method widely applicable.

5.5. *Hybrid EnVar PF*

Based on the localized adaptive particle filter (LAPF) described in Section 4.1.3, a hybrid particle filter based ensemble variational data assimilation system (PfVar) can also be constructed. The idea is to replace the LETKF-based ensemble in an EnVar by an LAPF-based ensemble.

We briefly discuss a practical numerical weather prediction example here. Following Buehner *et al.* (2013), the operational EnVAR system of DWD for the ICON model with 13km global resolution and 6.5km resolution of its two-way nested area over Europe is using the ensemble of the global 40 member LETKF for its dynamic covariance matrix with a ratio of 70:30 towards the classical NMC based covariance matrix of the three-dimensional variational data assimilation system with 3h cycling interval. The LETKF ensemble is replaced by the LAPF ensemble, where the quality control of the variational high-resolution run is used for the ensemble data assimilation system under consideration. In the current system, no recentering of the ensemble with respect to the variational mean estimator is carried out, leading to a form of weak coupling of the systems.

In a quasi-operational setup (without a high-resolution nest), the hybrid PfVAR is running stably for a period of one month. The observation minus background statistics show very promising behaviour in several case studies which are under investigation at DWD (Walter *et al.* 2018). In the current state of tuning, the forecast quality of the PfVAR seems comparable to the forecasts based on the LETKF-based EnVAR. These new results studied in combination with Robert *et al.* (2017) show that today's particle filters are approaching the quality of state-of-the-art operational ensemble data assimilation systems and are already becoming important tools on all scales of NWP.

5.6. *Discussion*

Hybrid particle-ensemble Kalman filter schemes, especially when implemented adaptively, can avoid weight collapse in the particle filter part of the hybrid in any situation. The price paid is that not all information from the observations is extracted when the posterior pdf is severely non-Gaussian, but in many situations this is not the dominant source of error. The reason why these schemes are competitive is that they do take into account some non-Gaussianity via the particle filter, while the particle filter alone is very inefficient compared to the Ensemble Kalman Filter when the posterior is actually close to a Gaussian. So the objective is not necessarily to make the α as small as possible, but indeed to find an optimal α to ensure that the Ensemble Kalman Filter is used whenever we can. The same is true for the bridging parameter in the Adaptive Gaussian Mixture Filter.

The second-order exact filters are hybrids of a different kind, focussing on obtaining the posterior mean and the covariance correct given the limited prior ensemble. These methods are expected to be quite competitive to the hybrid filters discussed above, and the relative performance will depend strongly on the measure used to define what is best. For instance, RMSE are expected to be better for the second-order exact filters, while full ensemble measures like rank histograms and continuous ranked probability scores might benefit from the hybrid schemes.

One question that emerges when comparing the Ensemble Kalman Particle Filter and the LETPF-LETKF hybrid is what should one use first, the particle filter or the ensemble Kalman filter. Different experimental results seem to indicate that both orderings can be superior to the other. The PF-first methods have the advantage of a theoretical justification via a two-step tempering interpretation in which the particle filter step makes the prior for the EnKF much more Gaussian. Applying the EnKF first will bring the particles closer to the observations, leading to better weight balance in the particle filter. At this moment it is unclear which order is best when, much more research is needed.

6. Conclusions and discussion

The largest issue of standard particle filters was until recently their degeneracy in high-dimensional settings: when the number of independent observations is large and the number of particles is limited (of order 10-1000 for geophysical applications), one particle gets weight one, and all others get weight zero.

Two developments have revived the interest in particle filters: efficient proposal densities and localisation, while hybrids with Ensemble Kalman Filters and recently transportation filters enhance confidence in the usefulness of particle filters in high-dimensional settings. The new kid on the block are particle flow methods. Their popularity in the large machine-learning community ensures rapid progress here, too. It is unclear at this moment how competitive these new ideas will be. It is clear that developments on particle filters have been very fast, and the first tests of both localised and hybrid particle-EnKF filters in operational numerical weather prediction have been performed and show highly encouraging results.

This paper discussed these new developments and demonstrates that particle filters are useful in even the largest dimensional geophysical data-assimilation problems and will allow us to make large steps towards fully nonlinear data assimilation. The emphasis was here on explaining and connecting existing and new ideas, including new understanding of the optimality of the optimal proposal density and equal-weight filters.

From the presentation it has become clear that the field is too young to provide solid guidance on which method will be most fruitful for which problem. Given that most data-assimilation practitioners will have an implementation of a local Ensemble Kalman Filter in some form, localised particle filters seem to be the fastest way to make progress. However, one has to keep in mind that the resampling step needs smoothing that is more complex than in an Ensemble Kalman Filter, although exciting new variants like the ETPF and LAPF allow for smooth updates in a very natural way. Furthermore, with the small ensemble sizes now practical (10-100), more than 10 independent observations in a localisation area may already lead to filter degeneracy, forcing us to look into methods that limit the weights from below. This is another ad-hoc procedure that limits information extraction from observations, but it is unclear how severe this issue is.

Even easier are implementations of hybrid PF-EnKF filters, but it is still unclear what these filters target. At the moment their value lies in bringing more non-Gaussianity into Ensemble Kalman Filters, but at the same time ensure that an Ensemble Kalman Filter is used when that is warranted.

We discussed two main variants that try to avoid localisation because of the issues discussed above: the equal-weight particle filters and transportation particle filters. The equal-weight variants, which avoid weight collapse by construction, do not have a complete mathematical foundation yet. We know these schemes are biased, but since they are tailored to high-dimensional problems with small ensemble sizes the bias error might be smaller than the Monte-Carlo error from the small ensemble size. Transportation particle filters still have to demonstrate their full potential in geoscience applications, but initial experiments with e.g. mapping particle filters on low-to-moderate dimensional systems together with the way they are formulated suggest they could become mainstream competitive schemes.

All in all, huge progress has been made in particle filtering, and initial attempts to implement the schemes into full-scale numerical weather prediction models have succeeded, with promising initial results. This shows that particle filters can no longer be ignored for high-dimensional geoscience applications.

Appendices

A. Law of total variance

The law of total variance is an elementary theorem in statistics and probability. It can be proven as follows. First we need the Law of total expectation, which reads, using $E_A[B]$ as denoting the expectation of B under pdf $p(a)$:

$$\begin{aligned} E_Y[E_{X|Y}[f(X)]] &= \int\int f(x)p(x|y)p(y)\;dx\;dy \\ &= \int_x\int_y f(x)p(x,y)\;dy\;dx \\ &= \int f(x)p(x)\;dx \\ &= E_X[f(X)] \end{aligned} \tag{143}$$

Using this equality on $var_{\mathbf{X}}[\mathbf{X}]$ leads to:

$$\begin{aligned} var_{\mathbf{X}}[\mathbf{X}] &= E_{\mathbf{X}}[\mathbf{X}^2] - E^2_{\mathbf{X}}[\mathbf{X}] \\ &= E_Y\left[E_{\mathbf{X}|Y}[\mathbf{X}^2]\right] - E^2_Y[E_{\mathbf{X}|Y}[\mathbf{X}]] \\ &= E_Y\left[var_{\mathbf{X}|Y}[\mathbf{X}] + E^2_{\mathbf{X}|Y}[\mathbf{X}]\right] - E^2_Y[E_{\mathbf{X}|Y}[\mathbf{X}]] \\ &= E_Y\left[var_{\mathbf{X}|Y}[\mathbf{X}]\right] + E_Y\left[E^2_{\mathbf{X}|Y}[\mathbf{X}]\right] - E^2_Y[E_{\mathbf{X}|Y}[\mathbf{X}]] \\ &= E_Y\left[var_{\mathbf{X}|Y}[\mathbf{X}]\right] + var_Y\left[E_{\mathbf{X}|Y}[\mathbf{X}]\right] \end{aligned} \tag{144}$$

which proves the theorem.

Acknowledgement

PJvL thanks the European Research Council (ERC) for funding the CUNDA project under the European Union Horizon 2020 research and innovation programme. The research of SR has been partially funded by Deutsche Forschungsgemeinschaft (DFG) through grant CRC 1294 “Data Assimilation (project A02)”. We thank 3 anonymous reviewers for their critical comments, leading to a much improved paper.

References

Ades M, van Leeuwen PJ. 2013. An exploration of the equivalent weights particle filter. *Q. J. R. Meteorol. Soc.* **139**: 820–840.

Ades M, van Leeuwen PJ. 2015a. The effect of the equivalent-weights particle filter on model balances in a primitive equation model. *Monthly Weather Rev.* **143**, doi:10.1175/MWR-D-14-00050.1.

Ades M, van Leeuwen PJ. 2015b. The equivalent-weights particle filter in a high-dimensional system. *Q. J. R. Meteorol. Soc.* **141**: 484–503.

Bengtsson T, Snyder C, Nychka D. 2003. Toward a nonlinear ensemble filter for high-dimensional systems. *J. Geophys. Res.* **108**: 8775–8785.

Beskos A, Crisan D, Jasra A. 2014. On the stability of sequential Monte Carlo methods in high dimensions. *Annals Applied Probab.* doi:10.1214/13-AAP951.

Beskos A, Crisan D, Jasra A, Kamatani K, Zhou Y. 2017. A stable particle filter for a class of high-dimensional state-space models. *Adv. Appl. Prob* **49**: 24–48, doi:10.1017.apr.2016.77.

Briggs J, Dowd M, Meyer R. 2013. Data assimilation for large-scale spatio-temporal systems using a location particle smoother. *Environmetrics* **24**: 81–97, doi:10.1002/env.2184.

Browne PA, van Leeuwen P. 2015. Twin experiments with the equivalent weights particle filter and hadcm3. *Q. J. R. Meteorol. Soc.* **141**, doi: 10.1002/qj.2621.

Buehner M, Morneau J, Charette C. 2013. Four-dimensional ensemble-variational data assimilation for global deterministic weather prediction. *Nonlinear Processes in Geophysics* **20**(5): 669–682, doi:10.5194/npg-20-669-2013.

Burgers G, van Leeuwen PJ, Evensen G. 1998. Analysis scheme in the ensemble Kalman filter. *Monthly Weather Review* **126**(6): 1719–1724.

Chen Y, Reich S. 2015. Assimilating data into scientific models: An optimal coupling perspective. In: *Frontiers in Applied Dynamical Systems: Reviews and Tutorials*, vol. 2, Springer-Verlag: New York, pp. 75–118.

Chorin AJ, Morzfeld M, Tu X. 2010. Interpolation and iteration for nonlinear filters. *Communications in Applied Mathematics and Computational Science* **5**: 221–240.

Chustagulprom N, Reich S, Reinhardt M. 2016. A hybrid ensemble transform filter for nonlinear and spatially extended dynamical systems. *SIAM/ASA J. Uncertainty Quantification* **4**: 592–608.

Daum F, Huang J. 2011. Particle flow for nonlinear filters. In: *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP 2011, May 22-27, 2011, Prague Congress Center, Prague, Czech Republic*. pp. 5920–5923, doi:10.1109/ICASSP.2011.5947709.

Daum F, Huang J. 2013. Particle flow for nonlinear filters, bayesian decisions and transport. In: *Proceedings of the 16th International Conference on Information Fusion, FUSION 2013, Istanbul, Turkey, July 9-12, 2013*. pp. 1072–1079, URL http://ieeexplore.ieee.org/document/6641115/.

de Wiljes J, Acevedo W, Reich S. 2017. Second-order accurate ensemble transform particle filters. *SIAM J. Sci. Comput.* **39**: A1834–1850.

Degond P, Mustieles FJ. 1990. A deterministic approximation of diffusion equations using particles. *SIAM J. Sci. Comput.* **11**: 293–310.

DelMoral P, Doucet A, Jasra A. 2006. Sequential Monte Carlo samplers. *J. R. Statist. Soc. B* doi:10.1111/j.1467-9868.2006.00553.x.

Doucet A, de Freitas N, Gordon N. 2001. *Sequential Monte Carlo methods in practice*. Springer.

Emerick A, Reynolds A. 2013. Ensemble smoother with multiple data assimilations. *Comput. Geosci.* **55**: 3–15.

Evensen G. 2018. Analysis of iterative ensemble smoothers for solving inverse problems. *Computational Geosciences* **22**.

Farchi A, Bocquet M. 2018. Review article: Comparison of local particle filters and new implementations. *Nonlin. Processes Geophys.* **25**: 765–807, doi:10.5194/npg-25-765-2018.

Frei M, Künsch HR. 2013. Bridging the ensemble Kalman and particle filters. *Biometrika* **100**: 781–800, doi:10.1093/biomet/ast020.

Hoteit I, Pham DT, Triantafyllou G, Korres G. 2008. A new approximate solution of the optimal nonlinear filter for data assimilation in meteorology and oceanography. *Mon. Wea. Rev.* **136**: 317–334.

Houtekamer PL, Mitchell HL. 1998. Data assimilation using an ensemble Kalman filter technique. *Mon. Wea. Rev.* **126**: 796–811.

Hunt BR, Kostelich EJ, Szunyogh I. 2007. Efficient data assimilation for spatiotemporal chaos: A local ensemble transform Kalman filter. *Physica D* **230**: 112–126.

Kirchgessner P, Toedter J, Ahrens B, Nerger L. 2017. The smoother extension of the nonlinear ensemble transform filter. *Tellus A* **69**: 1327 766.

Lei J, Bickel P. 2011. A moment matching ensemble filter for nonlinear non-Gaussian data assimilation. *Mon. Wea. Rev.* **139**: 3964–3973.

Liu Q, Wang D. 2016. Stein variational gradient descent: A general purpose Bayesian inference algorithm. *NIPS 2016* doi:arXiv:1608.04471.

Lu J, Lu Y, Nolen J. 2018. Scaling limit of the Stein variational gradient descent, Part I: The mean field regime. Technical Report arXiv:1805.04035, Duke University.

McCann R. 1995. Existence and uniqueness of monotone measure-preserving maps. *Duke Mathematical Journal* **80**: 309–323.

Morzfeld M, Hodyss D, Snyder C. 2017. What the collapse of the ensemble kalman filter tells us about particle filters. *Tellus A: Dynamic Meteorology and Oceanography* **69**.

Morzfeld M, Tu X, Atkins E, Chorin AJ. 2012. A random map implementation of implicit filters. *Journal of Computational Physics* **231**: 2049–2066.

Moselhy TE, Marzouk Y. 2012. Bayesian inference with optimal maps. *J. Comput. Phys.* **231**: 7815–7850.

Nakamura G, Potthast R. 2015. *Inverse modeling*. 2053-2563, IOP Publishing, ISBN 978-0-7503-1218-9, doi:10.1088/978-0-7503-1218-9.

Nakano S, Ueno G, Higuchi T. 2007. Merging particle filter for sequential data assimilation. *Nonlinear Processes Geophys.* **14**: 395–408.

Neal RM. 1996. Sampling from multimodal distributions using tempered transitions. *Statistics and Computing* **6**: 353–366.

Nerger L, Janjić T, Schroeter J, Hiller W. 2012. A unification of ensemble square root filters. *Mon. Wea. Rev.* **140**: 2335–2345.

Papadakis N, Memin E, Cuzol A, Gengembre N. 2010. Data assimilation with the weighted ensemble kalman filter. *Tellus A: Dynamic Meteorology and Oceanography* **62**.

Pathiraja S, Reich S. 2019. Discrete gradients for computational Bayesian inference. Technical Report arXiv:1903.00186, University of Potsdam.

Penny S, Miyoshi T. 2016. A local particle filter for high-dimensional geophysical systems. *Nonlinear Processes Geophys.* **23**(Nonlinear Processes Geophys.), doi:10.1175/MWR-D-15-0163.1.

Pham DT. 2001. Stochastic methods for sequential data assimilation in strongly nonlinear systems. *Mon. Wea. Rev.* **129**: 1194–1207.

Pitt MK, Shephard N. 1999. Filtering via simulation: Auxiliary particle filters. *Journal of the American Statistical Association* **94**(446): 590–599.

Poterjoy J. 2016. A localized particle filter for high-dimensional nonlinear systems. *Monthly Weather Rev.* **144**(Nonlinear Processes Geophys.), doi:10.1175/MWR-D-15-0163.1.

Poterjoy J, Anderson JL. 2016. Efficient assimilation of simulated observations in a high-dimensional geophysical system using a localized particle filter. *Monthly Weather Rev.* **144**(Nonlinear Processes Geophys.), doi:10.1175/MWR-D-15-0163.1.

Potthast R, Walter A, Rhodin A. 2019. A Localized Adaptive Particle Filter within an operational NWP framework. *Monthly Wea. Rev.* doi:10.1175/MWR-D-18-0028.1.

Pulido M, van Leeuwen P. 2018. Kernel embedding of maps for Bayesian inference: The variational mapping particle filter ArXiv:1805.11380.

Reich S. 2011. A dynamical systems framework for intermittent data assimilation. *BIT Numer Math* **51**: 235–249.

Reich S. 2012. A Gaussian mixture ensemble transform filter. *Q. J. R. Meterolog. Soc.* **138**: 222–233.

Reich S. 2013. A nonparametric ensemble transform method for Bayesian inference. *SIAM J. Sci. Comput.* **35**: A2013–A2024.

Reich S, Cotter C. 2015. *Probabilistic forecasting and bayesian data assimilation*. Cambridge University Press: Cambridge.

Robert CP, Cassela G. 2004. *Monte Carlo statistical methods*. Springer-Verlag.

Robert S, Künsch HR. 2017. Localizing the ensemble transform Kalman particle filter. *Tellus A* doi:10.1080/16000870.2017.1282016.

Robert S, Leuenberger D, Künsch HR. 2017. A local ensemble transform Kalman particle filter for convective scale data assimilation. *Q. J. R. Meterolog. Soc.* doi:10.1002/qj.3116.

Russo G. 1990. Deterministic diffusion of particles. *Comm. Pure Appl. Math.* **43**: 697–733.

Schraff C, Reich H, Rhodin A, Schomburg A, Stephan K, Periáñez A, Potthast R. 2016. Kilometre-scale ensemble data assimilation for the COSMO model (kenda). *Quarterly J. Roy. Meteorol. Soc.* **142**(696): 1453–1472, doi: 10.1002/qj.2748.

Snyder C, Bengtsson T, Bickel P, Anderson J. 2008. Obstacles to high-dimensional particle filtering. *Monthly Weather Review* **136**: 4629–4640.

Snyder C, Bengtsson T, Morzfeld M. 2015. Performance bounds for particle filters using the optimal proposal. *Monthly Weather Rev.* **143**, doi:10.1175/MWR-D-15-0144.1.

Spantini A, Bigoni D, Marzouk Y. 2017. Inference via low-dimensional couplings. ArXiv:1703.06131.

Stordal AS, Karlsen HA, Naevdal G, Skaug HJ, Valles B. 2011. Briding the ensemble Kalman filter and particle filters: the adaptive Gaussian mixture filter. *Comput. Geosci.* **15**: 293–305.

Tödter J, Ahrens B. 2015. A second-order exact ensemble square root filter for nonlinear data assimilation. *Mon. Wea. Rev.* **143**(4): 1347–1367.

Tödter J, Kirchgessner P, Nerger L, Ahrens B. 2016. Assessment of a nonlinear ensemble transform filter for high-dimensional data assimilation. *Mon. Wea. Rev.* **144**: 409–427.

van Leeuwen PJ. 2003. Nonlinear ensemble data assimilation for the ocean. In: *Recent Developments in data assimilation for atmosphere and ocean, ECMWF Seminar 8-12 September 2003, Reading, United Kingdom*. pp. 265–286.

van Leeuwen PJ. 2009. Particle filtering in geophysical systems. *Mon. Wea. Rev.* **137**: 4089–4114.

van Leeuwen PJ. 2010. Nonlinear data assimilation in Geosciences: An extremely efficient particle filter. *Q. J. R. Meteorol. Soc.* **136**: 1991–1999.

van Leeuwen PJ. 2015. Representation errors and retrievals in linear and nonlinear data assimilation. *Q.J.Royal. Meteorol.Soc.* **141**: 1612–1623, doi: 10.1002/qj.2464.

van Leeuwen PJ, Cheng Y, Reich S. 2015. *Nonlinear data assimilation*. Springer, doi:10,1007/978-3-319-18347-3.

Vetra-Carvalho S, van Leeuwen PJ, Nerger L, Barth A, Altaf MU, Brasseur P, Kirchgessner P, Beckers JM. 2018. State-of-the-art stochastic data assimilation methods for high-dimensional non-gaussian problems. *Tellus A: Dynamic Meteorology and Oceanography* **70**, doi:10.1080/16000870.2018.1445364.

Villani C. 2008. *Optimal transport: Old and new*. Springer Science & Business Media: New York.

Vossepoel FC, van Leeuwen PJ. 2006. Parameter estimation using a particle method: Inferring mixing coefficients from sea-level observations. *Monthly Weather Rev.* **135**: 1006–1020.

Walter A, Potthast R, Rhodin A. 2018. On hybrid particle-filter based ensemble variational data assimilation URL `inPreparation`.

Xiong X, Navon IM, Uzunoglu B. 2006. A note on the particle filter with posterior Gaussian resampling. *Tellus* **58A**: 456–460.

Zhu M, van Leeuwen PJ, Amezcua J. 2016. Implicit equal-weights particle filter. *Q. J. R. Meteorol. Soc.* doi:10.1002/qj.2784.