SANGOMA: Stochastic Assimilation for the Next Generation Ocean Model Applications EU FP7 SPACE-2011-1 project 283580

Deliverable 3.4:
Report on new methods
Due date: 31/5/2015
Delivery date: 31/10/2015
Delivery type: Report, public

Peter Jan Van Leeuwen   Sanita Vetra-Carvalho
University of Reading, UK

Lars Nerger   Paul Kirchgessner
Alfred-Wegener-Institute, GERMANY

Arnold Heemink   Nils van Velzen
Martin Verlaan   M. Umer Altaf
Delft University of Technology, NETHERLANDS

Jean-Marie Beckers   Alexander Barth
University of Liège, BELGIUM

Pierre Brasseur   Jean-Michel Brankart   Guillem Candille
CNRS-LEGI, FRANCE

Pierre De Mey
CNRS-LEGOS, FRANCE

Laurent Bertino   Alberto Carrassi
NERSC, NORWAY

Contents

Introduction

Important part of work package 3 of SANGOMA project is development of new data assimilation methods and their implementation in Matlab/Octave and/or Fortran languages compliant with specifications in work package 1. Each of the new methods have been described in the living document, deliverable D3.2, which is updated through the project and is presented in it's final form as deliverable D3.5. This report contains the documentation and description of Matlab/Octave and/or Fortran codes for the methods which have been developed throughout SANGOMA project.

The new methods which implementation will be discussed in this report are:

- Equal Weights Particle filter or EWPF from UREAD.
- Implicit Equal Weights Particle Filter or IEWPF from UREAD.
- Nonlinear Ensemble Transform Filter or NETF from AWI.
- Multivariate Rank Histogram Filter or MRHF from CNRS/LGGE.

Implementation of EWPF (UREAD)

2.1 Description of EWPF method

The Equivalent Weights Particle Filter [Van Leeuwen, 2010, Ades and van Leeuwen, 2012] or EWPF nudges particles towards future observations at the forecast step and picks significant particles from the posterior density at the analysis step with weights that are very close to each other.

The full description of the EWPF method can be found in Sangoma deliverable D3.1 as well as the final Sangoma deliverable D3.5.

2.2 Description of the EWPF code

The EWPF is the one common new data assimilation method that has been implemented across most toolboxes in Sangoma project including EMPIRE. in UREAD there are two versions of EWPF that have been implemented:

- Original EWPF code in FORTRAN and PYTHON as developed and implemented in EMPIRE by Browne and Wilson [2015].
- Sangoma EWPF code in FORTRAN and MATLAB which has been adapted to be complaint with Sangoma data models and specifications in work package WP1.

The codes are very similar with the Sangoma version being a clean and easily portable as well as C-bound version of the original EMPIRE code. The original EWPF code in EMPIRE is open source and can be downloaded from:
www.met.reading.ac.uk/ darc/empire/doc/html/.

Sangoma EWPF code in FORTRAN and MATLAB are both also open source and can be downloaded from
sourceforge.net/p/sangoma/code/HEAD/tree/workpackages/wp3/EWPF_code/.

The full Sangoma code structure and implementation in EMPIRE (as well as other Sangoma toolboxes) can be found in Sangoma deliverable D3.3.

2.3 Implementation of the code in the EMPIRE

The EWPF has been implemented into EMPIRE (Employing MPI for Researching Ensembles) and full details, including EMPIRE code in FORTRAN, are available online www.met.reading.ac.uk/ darc/empire/doc/html/.

There are a number of particle and ensemble filters implemented in EMPIRE. A user can select the type of filter to be used by changing settings in 'pf_parameters.dat'. In particular for the EWPF, one has to select 'filter = 'EW". The 'pf_parameters.dat' data file is also used to select other options for the filter such as observation generation, number of observations (if generated), run twin experiments or not. In addition, some of the parameters in 'user_base.F90' are defined in this file such as 'efac', 'ufac', 'keep', and 'nudgefac'.

When run, the first thing EMPIRE does is to call the 'configure_model' routine which returns the dimensions of the state and next set of observations as well as the time at which next observations are available. However, a user can and needs to use the 'configure_model' routine to load in memory other data such as data necessary for call-back routines 'cb_Qhalf' and 'cb_solve_r'. Further, if $\mathbf{HQH}^T + \mathbf{R}$ does not change over time then the user also can factorise it in 'configure_model' to save computational time later. After the assimilation of the first observations has been completed, EMPIRE calls 'reconfigure_model' which returns the time at which next observations are available and their dimension. This continues until all observations are assimilated.

2.4 Benchmarks tests

Both EWPF versions have been run and tested with the small benchmark Lorenz96 model [Lorenz and Emmanuel, 1998] with 40 variables.

The original EWPF code in EMPIRE has recently been coupled with the medium Sangoma benchmark NEMO GYRE, however, results are not yet available but we aim to have results in a very near future.

Sangoma EWPF code both in FORTRAN and MATLAB were tested using the Lorenz 96 model with 40 variables of which first 20 were observed with linear H, and last 20 were not observed. We used 25 particles with 1000 model timesteps and 100 observation sets equally spaced in time. The initial condition for the reference solution is spun up from null state for 1000 timesteps. We used a very simple diagonal observation and model error covariances with standard deviations of $\sigma_R = 1.0$ and $\sigma_Q = 0.5$, respectively. The initial ensemble was created around the initial reference solution with standard deviation $\sigma_E = 0.2$. Figure 2.1 is showing results for x_1 variable in the model with the reference solution in blue, observations as black dots and ensemble mean in dashed red in the upper plot while the lower plot shows the ensemble standard deviation and RMSE between ensemble mean and truth.

Figure 2.2 is showing results for x_{10} variable with the same legends. It is clearly seen that the EWPF for x_1 has a greater difficulty tracking solution than for variable x_{10} since x_1 has no information coming from the left since neither x_{40} nor x_{39} are observed while x_{10} has ten preceding observed variables.

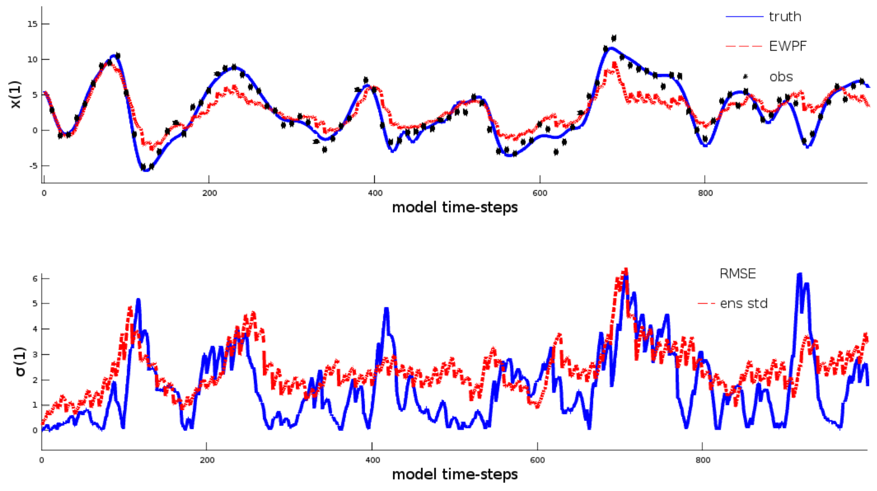

Figure 2.1: Lorenz 96 with EWPF for x_1 variable.

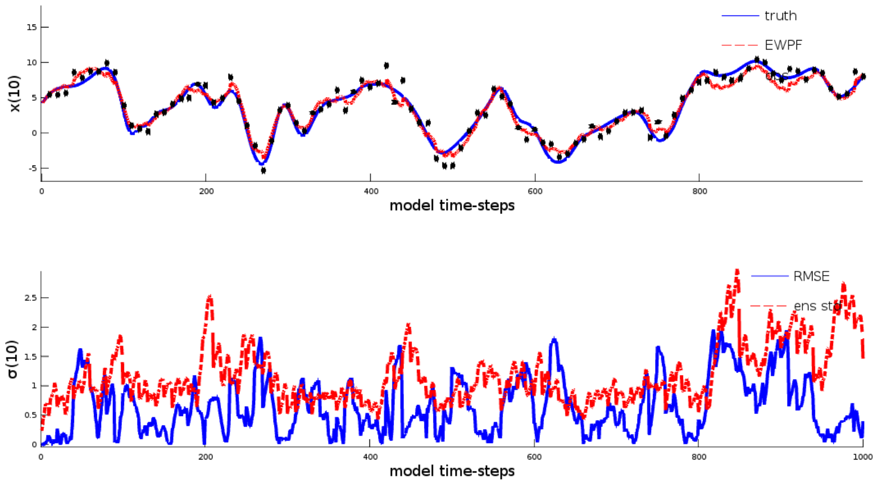

Figure 2.2: Lorenz 96 with EWPF for x_{10} variable.

Implementation of IEWPF (UREAD)

3.1 Description of IEWPF method

The Implicit Equivalent Weights Particle filter or IEWPF has been recently developed by Zhu et al. [2015] and is similar to EWPF; however, instead of drawing sample directly from the proposal density as is done in the EWPF method, here they are drawn implicitly. The full description of the method can be found in the final Sangoma deliverable D3.5 as well as the original paper Zhu et al. [2015].

3.2 Description of the IEWPF code

The IEWPF is implemented in EMPIRE and the open source code can be downloaded from www.met.reading.ac.uk/ darc/empire.

3.3 Implementation of the code in the EMPIRE

Implementation of the IEWPF is very similar to the original EWPF in EMPIRE. While the code is in FORTRAN 90/95 and uses basic data types such as arrays and matrices, the code is not strictly compliant with work package WP1 specifications since it does not:

- use 'sangoma_base' module to specify precision of the variables;
- use C-binding.

3.4 Benchmark tests

The IEWPF has been tested on small benchmark Lorenz 96 with 1000 dimensions. Figures 3.1 and 3.2 show performance of the IEWPF method for observed variable x_{856} and its rank histogram correspondingly (figures taken from Zhu et al. [2015], personal communication). Similarly figures 3.3 and 3.4 show performance of the IEWPF method for observed variable x_{856} and its rank histogram correspondingly (figures after Zhu et al. [2015], personal communication).

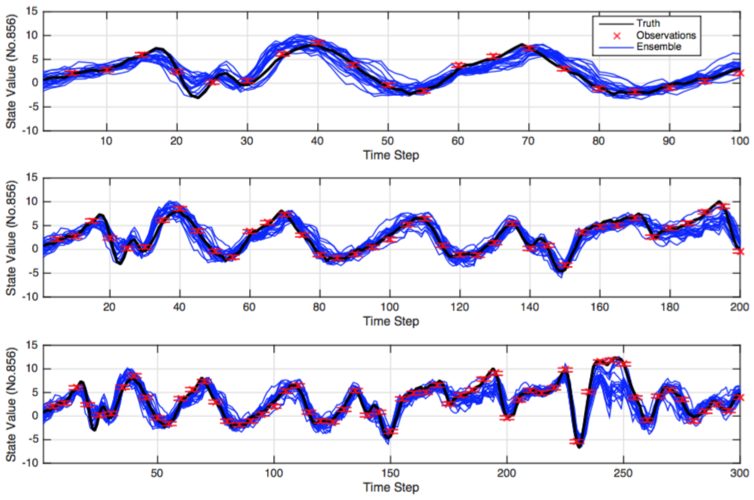

Figure 3.1: 1000 dimensional Lorenz 96 with IEWPF for observed variable x_{856}. The black line is the truth, the blue lines depict the evolution of the particles and red crosses are observations. (Figure after Zhu et al. [2015], personal communication)

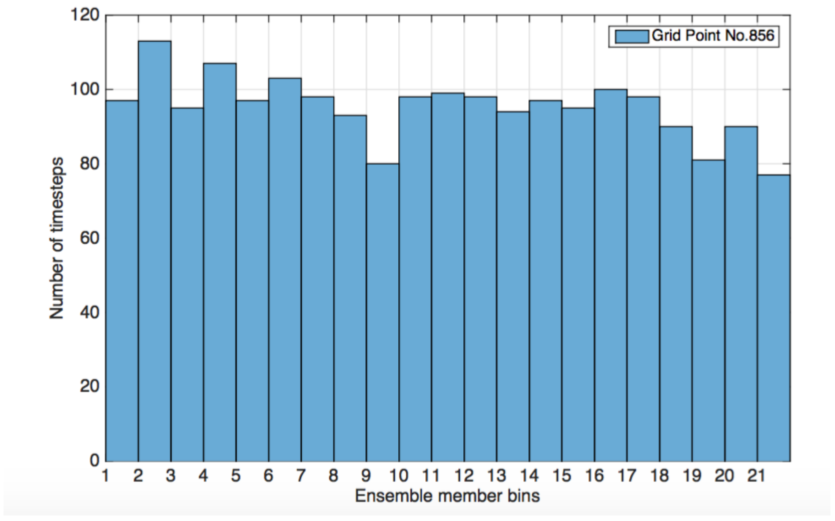

Figure 3.2: Rank histogram for variable x_{856} after a 10,000 time-steps model running for IEWPF with 1000 dimensional Lorenz 96 model. (Figure after Zhu et al. [2015], personal communication)

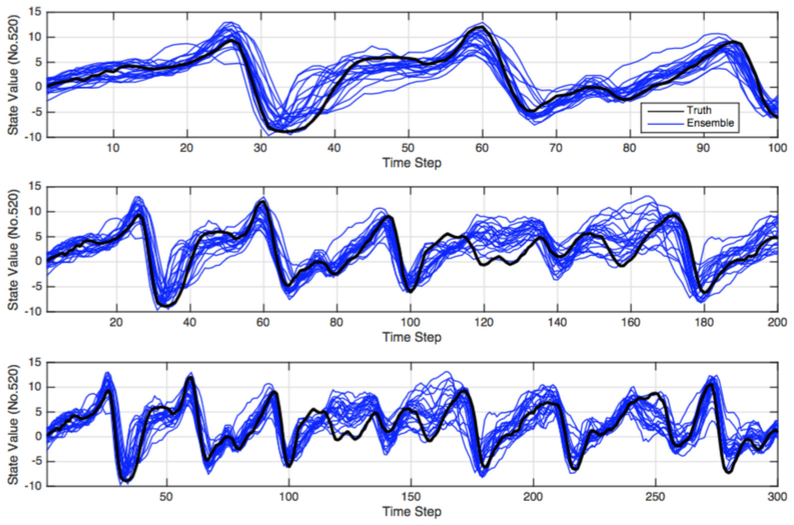

Figure 3.3: 1000 dimensional Lorenz 96 with IEWPF for unobserved variable x_{520}. The black line is the truth, the blue lines depict the evolution of the particles and red crosses are observations. (Figure after Zhu et al. [2015], personal communication)

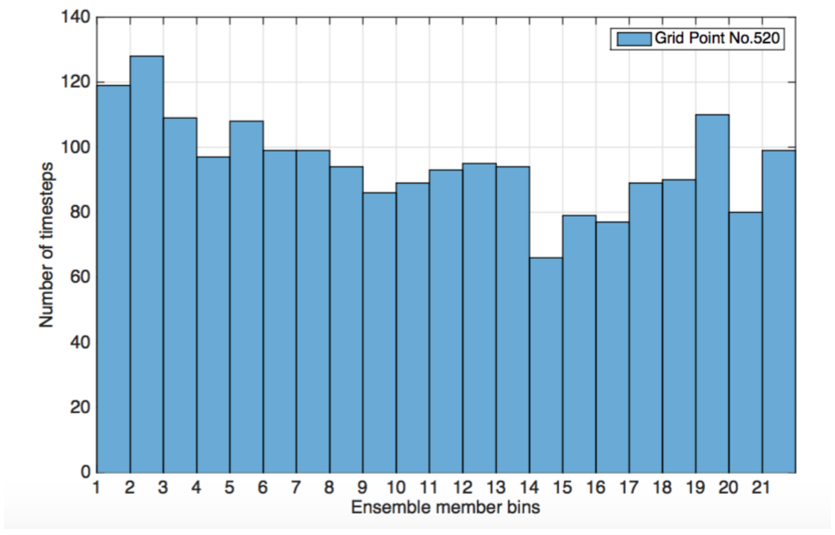

Figure 3.4: Rank histogram for variable x_{520} after a 10,000 time-steps model running for IEWPF with 1000 dimensional Lorenz 96 model. (Figure after Zhu et al. [2015], personal communication)

Implementation of NETF (AWI)

4.1 Description of NETF method

The nonlinear ensemble transform filter (NETF), which was recently introduced in [Tödter and Ahrens, 2015] was implemented in PDAF and further tested using the medium case benchmark.

While the EnKFs are based on a Gaussian assumption for the prior ensemble, the transformation in the NETF is designed to exactly match the first two moments of Bayes theorem [see Tödter and Ahrens, 2015, for a derivation of the NETF].

The ensemble mean at time t_k is computed as $\overline{\mathbf{x}}_k = \frac{1}{m}\mathbf{X}_k\mathbf{1}$, using the vector $\mathbf{1} = (1, \ldots, 1)^T$ of length m. Defining the following matrix,

$$\overline{\mathbf{X}}_k = \left[\overline{\mathbf{x}}_k, \ldots, \overline{\mathbf{x}}_k\right] = \frac{1}{m}\mathbf{X}_k\mathbf{1}\mathbf{1}^T, \tag{4.1}$$

the ensemble perturbation matrix $\mathbf{X}'_k$ is given by

$$\mathbf{X}'_k = \mathbf{X}_k - \overline{\mathbf{X}}_k = \mathbf{X}_k\mathbf{S}, \qquad \text{where} \qquad \mathbf{S} = \mathbf{I}_m - \frac{1}{m}\mathbf{1}\mathbf{1}^T. \tag{4.2}$$

Here, $\mathbf{I}$ denotes the identity matrix. Hence, the matrix $\mathbf{S}$ subtracts the ensemble mean from $\mathbf{X}_k$.

The NETF transforms the forecast ensemble into an analysis ensemble by applying a weight vector and a transform matrix to the forecast mean and perturbations, respectively. As most particle filters, the NETF uses the likelihood weights that arise from Bayes theorem. For normally distributed observation errors, the weight of each member is given by

$$w^i \propto \mathcal{N}\left(\mathbf{y}; \mathcal{H}\left(\mathbf{x}_k^{f(i)}\right), \mathbf{R}_k\right) \tag{4.3}$$

$$\propto \exp\left[-\frac{1}{2}\left(\mathbf{y} - \mathbf{y}_k^{f(i)}\right)^T \mathbf{R}_k^{-1}\left(\mathbf{y} - \mathbf{y}_k^{f(i)}\right)\right], \tag{4.4}$$

where $\mathbf{y}_k^{f(i)} = \mathcal{H}_k(\mathbf{x}_k^{f(i)})$. The weights are normalized so that they sum up to one. Before the weights are computed, the ensemble perturbations should be inflated by an inflation factor $\gamma > 1$. The weight vector and transform matrix of the NETF are computed from these weights as follows [Tödter and Ahrens, 2015]:

$$\mathbf{w} = (w^1, \ldots, w^m)^T \tag{4.5}$$

$$\mathbf{T} = \sqrt{m}\left[\mathbf{Diag}(w) - \mathbf{w}\mathbf{w}^T\right]^{1/2}. \tag{4.6}$$

Here, $\mathbf{Diag}(w)$ is a diagonal matrix that contains the weights w^i on the diagonal. To complete the algorithm, the analysis perturbations are computed as

$$\mathbf{X}_k^{a'} = \mathbf{X}_k^{f'}\mathbf{T}\mathbf{\Lambda}, \tag{4.7}$$

where $\mathbf{\Lambda}$ is an random orthogonal matrix [see Pham, 2001]. Finally, the mean is updated via

$$\overline{\mathbf{X}}_k^a = \overline{\mathbf{X}}_k^f + \mathbf{X}_k^{f'}\mathbf{w}\mathbf{1}^{\mathbf{T}}, \tag{4.8}$$

Thus, the new analysis ensemble is given by

$$\mathbf{X}_k^a = \overline{\mathbf{X}}_k^a + \mathbf{X}_k^{a'}. \tag{4.9}$$

The NETF has to be applied in combination with localization as described in [Tödter and Ahrens, 2015] for high-dimensional systems.

4.2 Description of the NETF code

INTERFACE:

```
SUBROUTINE sangoma_netf_analysis(dim, dim_obs, dim_ens, ens, &
     Hens, obs, forget, cb_prodRinvA, flag) &
     BIND(C, name="sangoma_netf_analysis_")
```

DESCRIPTION:

Analysis step of the NETF (Nonlinear Ensemble Transform Filter) following J. Toedter and B. Ahrens, A Second-order Exact Ensemble Square Root Filter for Nonlinear Data Assimlation, Mon. Wea. Rev. 143 (2015) 1347-1367.

This code is a simplified version of the NETF implemented in PDAF. E.g. the parallelization, memory counting and timers have been removed. Also the interface is shortened and automatic arrays are used instead of dynamic allocation. The observation operator is applied outside of the analysis routine, which leads to a restriction to linear observation operators. The routine uses a blocked ensemble transformation as in PDAF to avoid the need for a second ensemble array.

REVISION HISTORY:

```
2014-03 - Paul Kirchgessner - Changed original PDAF-ETKF code to NETF
2015-10 - Lars Nerger - Adaption for Sangoma
```

USES:

```
USE, INTRINSIC :: ISO_C_BINDING
USE sangoma_base, ONLY: REALPREC, INTPREC
USE mod_sangoma_analysis, ONLY: generate_rndmat

IMPLICIT NONE
```

ARGUMENTS:

```
INTEGER(INTPREC), INTENT(in) :: dim_obs   ! Dimension of observation
      vector
INTEGER(INTPREC), INTENT(in) :: dim_ens   ! Size of ensemble
REAL(REALPREC), INTENT(inout) :: ens(dim, dim_ens) ! State ensemble
      – overwritten
                                          ! In: forecast; Out: analysis
                                               ensemble
REAL(REALPREC), INTENT(inout) :: Hens(dim_obs, dim_ens) ! Observed
      ensemble
REAL(REALPREC), INTENT(in) :: obs(dim_obs) ! Vector of observations
REAL(REALPREC), INTENT(in) :: forget      ! Forgetting factor
INTEGER(INTPREC), INTENT(out) :: flag     ! Status flag

! External  call–back subroutine
INTERFACE
   ! Routine computing the product of inverse observation error covariance
      matrix
   ! with some other matrix, which is a temporary result in the ETKF
   SUBROUTINE cb_prodRinvA(dim_obs, dim_ens, HZ, RiHZ) BIND(C)
     USE sangoma_base, ONLY: REALPREC, INTPREC
     INTEGER(INTPREC), INTENT(in) :: dim_obs            ! Observation
         dimension
     INTEGER(INTPREC), INTENT(in) :: dim_ens            ! Ensemble
         size
     REAL(REALPREC), INTENT(in) :: HZ(dim_obs, dim_ens) ! Input matrix
          in observation space
     REAL(REALPREC), INTENT(out) :: RiHZ(dim_obs, dim_ens) ! The
         result of the multiplication
   END SUBROUTINE cb_prodRinvA
END INTERFACE
```

4.3 Implementation of the code in the PDAF

Since algorithmically the only difference between the ETKF and the NETF is the computation of the transform matrix, the code of the NETF is directly based on the existent ETKF code. The computation of the particle weights is changed, so that the computation is based on the particle weights, as defined in the chapter above. Although, the NETF allows for arbitrary distributed observations, in the implementation of PDAF an Gaussian distribution was assumed, since most often this is used.

4.4 Benchmark tests

The NETF was previously applied to several different small sized models [see Tödter et al., 2015]. Recently, using the medium case benchmark, Tödter et al.

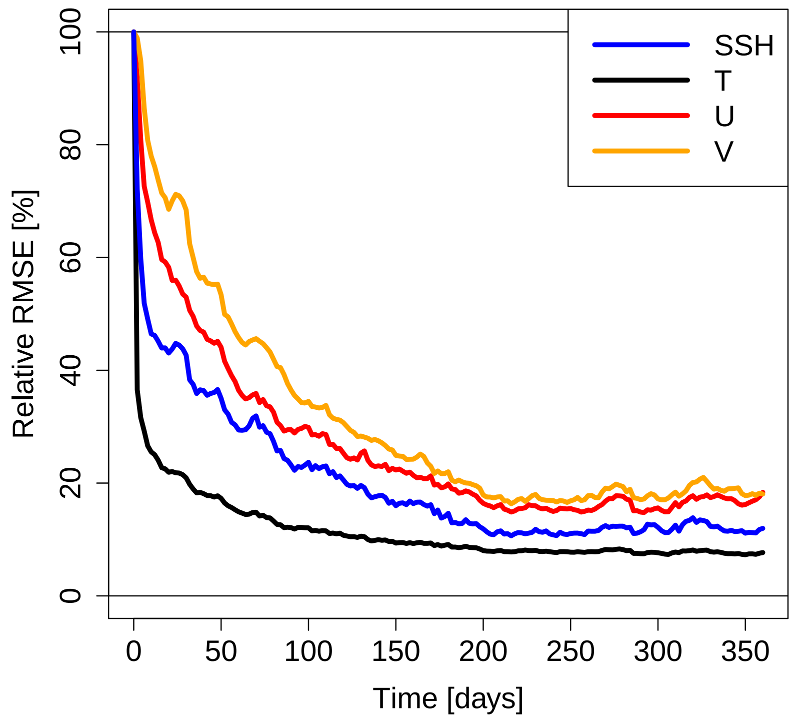

Figure 4.1: RMSE of the NETF in the medium case benchmark, relative to the error at the first analysis time. (Image taken from [Tödter et al., 2015])

[2015] demonstrated that the NETF is also applicable to high-dimensional assimilation problems with an ensemble size that is computationally feasible.

Using 120 ensemble members the errors in the analysis in this setup could be reduced below than 10% after one year of assimilation (see. Fig. 4.1). The reduction is largest in the observed fields SSH and T. However, the errors in the unobserved fields are also reduced to about 20%. Additional details to the results can be found in [Tödter et al., 2015].

Implementation of MRHF (CNRS/LGGE)

5.1 Description of MRHF method

The Multivariate Rank Histogram Filter (MRHF) has been developed by Metref et al. [2014]. It is a fully non-Gaussian method, like standard particles filters. It is a *transform* method, like all the methods based on the Kalman filter, and contrary to standard particle filters that are *sampling* methods. For this reason, it is expected to be more resilient to ensemble collapse than standard particle filters. Localisation can be applied with the MRHF similarly to Ensemble Kalman Filters. Finally, the MRHF is quite flexible and can be implemented in connexion with an EnKF for example, to analyse only a few observations or a few variables of a specific type. The description of the method can be found in the final Sangoma deliverable D3.5 as well as the original paper Metref et al. [2014].

5.2 Description of MRHF code

The MRHF is still under development to reduce its computational complexity. These developments are performed in PYTHON which provides a flexible environment with many built-in functions useful for the method (sorting, searching, etc). Developments are still needed to confirm its compliance to high-dimensional systems that would motivate its conversion in a more computationally efficient language.

5.3 Benchmark tests

The MRHF has been implemented with the Lorenz 63 system [Lorenz, 1963] and a 1D realistic ecosystem model (see deliverable D4.4). Tests have also been conducted with the small benchmark Lorenz 96, though not utterly finalised. Figure 5.1 shows a comparison of Root Mean Square Errors from data assimilation experiments performed with the Lorenz 63 system, using different assimilation methods (differentiated by the different lines), different ensemble sizes (horizontal axes), and three observation frequencies: every 10, 25, and 50 time steps (the three panels). All three variables are observed with an error variance of 2. These setups correspond to mildly, moderately and strongly nonlinear test cases, respectively. The fully non-Gaussian methods (MRHF and particle filter) are more efficient than the others for moderately and strongly non-Gaussian situations. In the strongly nonlinear case, the MRHF performs much better than the particle filter with moderate ensemble sizes. These is encouraging for the perspective of using the MRHF as a fully non-Gaussian method with high-dimensional systems.

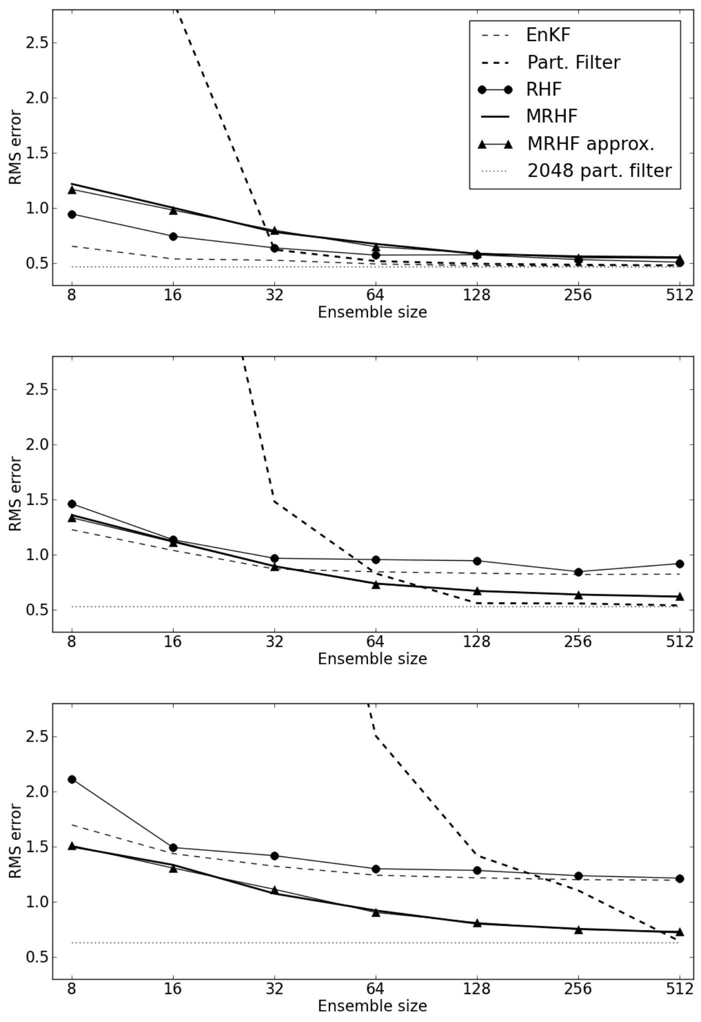

Figure 5.1: Time-averaged analysis RMSE for the EnKF (dashed line), the RHF (line with circles), the particle filter (with resampling, thick dashed line), the full MRHF (thick solid line) and an approximated MRHF (line with triangles); for experiments on the fully observed Lorenz 63 with observation time intervals $\Delta t = 0.10$ (upper-panel), $\Delta t = 0.25$ (center-panel) and $\Delta t = 0.50$ (bottom-panel). The dotted-line represents the time-averaged analysis RMSE for the particle filter with 2048 particles, which can be considered as a target score.

Local anamorphosis transformation (CNRS/LGGE)

6.1 Description of the local anamorphosis method

The basic problem of the algorithm is to look for a nonlinear change of variable transforming a random variable X with known cumulative distribution function (cdf) $F(x) = p(X \leq x)$ into a new random variable Z with the target cdf $G(z) = p(Z \leq z)$. Elementary probability calculus provides a general solution for the forward and backward transformations:

$$Z = G^{-1}\left[F(X)\right] \quad \text{and} \quad X = F^{-1}\left[G(Z)\right] \tag{6.1}$$

providing that F and G are invertible. In particular, if $Z \sim \mathcal{U}(0,1)$ is uniformly distributed on the interval $[0,1]$, with $G(z) = z$, then $x = F^{-1}(k/q)$ is the kth q-quantile of X; and if $Z \sim \mathcal{N}(0,1)$ is normally distributed, with $G(z) = \frac{1}{2}[1 + \text{erf}(z/\sqrt{2})]$, then Eq. (6.1) defines the forward and backward Gaussian anamorphosis transformation of the random variable X (Wackernagel, 2003, chapter 33).

Efficient approximate algorithm. In the Monte Carlo estimation methods (like the ensemble Kalman filter), the prior probability distribution for the control variables is only approximately described by a finite-size sample. The anamorphosis transformation in Eq. (6.1) for each control variable can thus only be approximately computed from the available sample using a nonparametric estimate $\tilde{F}(x)$ of the exact marginal cdf $F(x)$. The most simple nonparametric estimate of a probability density function (pdf) $\tilde{f}(x) = d\tilde{F}(x)/dx$ is the histogram: a piecewise constant pdf $\tilde{f}(x)$, or a piecewise linear cdf $\tilde{F}(x)$. As a simple choice for the classes of the histogram, we may use prescribed quantiles $\tilde{x}_k,\ k = 1,\ldots,q$ of the input sample, i.e. such that $\tilde{F}(\tilde{x}_k) = r_k$, for a given set of r_k $(0 \leq r_k \leq 1$, $r_k < r_{k+1})$. In this way, we can control explicitly the fraction of ensemble members $(r_{k+1} - r_k)$ in each class of the histogram.

Then, with the same level of approximation, we can use the same histogram representation of the Gaussian distribution, i.e. a piecewise linear $\tilde{G}(z)$ interpolating the true Gaussian cdf between $G(z_k) = r_k,\ k = 1,\ldots,q$, so that the anamorphosis transformation in Eq. (6.1) is also piecewise linear:

$$\varphi_{\text{forw}}(x) = \tilde{G}^{-1}\left[\tilde{F}(x)\right] = z_k + \frac{z_{k+1} - z_k}{\tilde{x}_{k+1} - \tilde{x}_k}(x - \tilde{x}_k) \quad \text{for} \quad x \in [\tilde{x}_k, \tilde{x}_{k+1}] \tag{6.2}$$

$$\varphi_{\text{back}}(z) = \tilde{F}^{-1}\left[\tilde{G}(z)\right] = \tilde{x}_k + \frac{\tilde{x}_{k+1} - \tilde{x}_k}{z_{k+1} - z_k}(z - z_k) \quad \text{for} \quad z \in [\tilde{z}_k, \tilde{z}_{k+1}]. \tag{6.3}$$

This approximate transformation remaps the quantiles $\tilde{x}_k,\ k = 1,\ldots,q$ of the input sample on the corresponding Gaussian quantiles $z_k,\ k = 1,\ldots,q$, and interpolates linearly between them. It is bijective between the interval $[\tilde{x}_1, \tilde{x}_q]$ and $[z_1, z_q]$, providing that the quantiles $\tilde{x}_k$ are distinct: $\tilde{x}_k \neq \tilde{x}_{k+1}\ \forall k$.

A full description of this algorithm, with many examples, can be found in Brankart et al. [2012].

6.2 Description of the local anamorphosis code

The anamorphosis code is available in the SANGOMA toolbox, using the standard SANGOMA coding rules:

INTERFACE:

```
SUBROUTINE sangoma_anamorphosis(dim, dim_qua, dir, &
    qua_ref, qua, vct, status) &
    BIND(C, name="sangoma_anamorphosis_")
```

DESCRIPTION:

This routine performs local anamorphosis transformation of a state or observation vector using a set of quantiles of the input ensemble (as provided by `sangoma_ComputeQuantiles`). This is an implementation of the specific algorithm described in Brankart et al. [2012].

Inputs are the vector to transform, the quantiles of the ensemble, and the corresponding quantiles of the target distribution (assumed all distinct).

REVISION HISTORY:

```
2015-02 - J.-M. Brankart - Initial SANGOMA code
```

USES:

```
USE, INTRINSIC :: ISO_C_BINDING
USE sangoma_base, ONLY: REALPREC, INTPREC

IMPLICIT NONE
```

ARGUMENTS:

```
INTEGER(INTPREC), INTENT(in)  :: dim  ! Vector (state or obs) dimension
INTEGER(INTPREC), INTENT(in)  :: dim_qua  ! Number of quantiles
INTEGER(INTPREC), INTENT(in)  :: dir    ! Forward (+) or backward (-) transform.
REAL(REALPREC), INTENT(in)    :: qua_ref(dim_qua) ! Quantiles of the target distribution
REAL(REALPREC), INTENT(in)    :: qua(dim, dim_qua) ! Ensemble quantiles
REAL(REALPREC), INTENT(inout) :: vct(dim)  ! Vector to transform
INTEGER(INTPREC), INTENT(out) :: status  ! Status flag (0=success)
```

COMMENT:
For an example on using `sangoma_Anamorphosis` see `example_Anamorphosis`.

6.3 Implementation of the code in SESAM

The SANGOMA code is a rewriting of a code previously developed in the SESAM software, as documented in Brankart et al. [2012]. With SESAM, the algorithm can be implemented as follows:

1. *Computation of the quantiles of the input ensemble*, with the SESAM command line:

   ```
   sesam -mode anam  -inxbas [ens_dir]
                     -outxbasref [quant_dir]
   ```

   where *[ens_dir]* is a directory containing the input ensemble forecast (as a set of NetCDF files, using the SESAM naming conventions), and *[quant_dir]*, a directory containing as an input, the definition of the quantiles (an ASCII file with the $r_k,\ k = 1, \dots, q$). From this, SESAM computes the (local) quantiles of the ensemble $\tilde{x}_k,\ k = 1, \dots, q$ (as a set of NetCDF files, in the directory *[quant_dir]*), linearly interpolating between successive ensemble members, if necessary.

2. *Local anamorphic transformation of the input ensemble*, with the SESAM command line:

   ```
   sesam -mode anam  -inxbas [ens_dir]
                     -inxbasref [quant_dir]
                     -outxbas [aens_dir]
                     -typeoper +
   ```

   where *[ens_dir]* is a directory containing the input ensemble forecast, and *[quant_dir]*, a directory containing the quantiles $\tilde{x}_k,\ k = 1, \dots, q$ of the ensemble (as obtained from the previous tool), and, as an additional input, the quantiles of the target distribution (an ASCII file with the $z_k,\ k = 1, \dots, q$). From this, SESAM computes the transformed ensemble (as a set of NetCDF files, in the directory *[aens_dir]*), by linearly interpolating between the z_k using Eq. (6.2). In this way, the transformation can easily be performed towards any target distribution (by just changing the ASCII file with the z_k), in particular towards the Gaussian distribution. (The backward transformation of Eq. (6.3) can be performed similarly by replacing the + sign by a – sign in the command line.)

Hence, only two SESAM command lines are sufficient to apply the algorithm to a variety of oceanographic applications. The first one (1) provides the histogram description of the marginal uncertainties. This is used by the second one (2) to perform the piecewise linear local anamorphic transformation, as a preprocessing to any operation taking profit from Gaussianity.

6.4 Benchmark tests

The above described anamorphosis algorithm has been tested and evaluated using a wide set of oceanographic applications as described in Brankart et al. [2012]. Here is one of these examples using the ecosystem component of the large scale SANGOMA benchmark.

In this example, we study the stochastic response of a coupled physical-biogeochemical model (CPBM) of the North Atlantic to uncertainties in the wind forcing. For that purpose, we use a 200-member designed as follows:

1. the CPBM couples a $1/4°$ resolution circulation model of the North Atlantic with the LOBSTER biogeochemical model, with 6 prognostic variables in the euphotic layer: phytoplankton (PHY), zooplankton (ZOO), nitrate (NO_3), ammonium, detritus, and semi-labile dissolved organic nitrogen;

2. the ensemble forecast is initialised at the beginning of the spring bloom on April 15, 1998; and

3. the random wind perturbations are sampled from a Gaussian probability distribution, with zero mean and a covariance derived from the ERA40 variability.

In the ensemble forecast, the main impact of the random wind perturbations on the ecosystem results from the deepening and shallowing of the mixed layer, which modifies the nutrient supply and thus the primary production in the euphotic layer. This mechanism produces a quite heterogenous response in terms of phytoplankton concentration, as illustrated in Fig. 6.1 by three deciles of the ensemble (corresponding to $r_k = 0.2$, 0.5 and 0.8, top panels) and one of the ensemble members (bottom panels). The wind can indeed only trigger a large ensemble dispersion (i.e. large differences between the deciles, in the top panels) in areas where the spring bloom has already started, like primarily the Gulf Stream pathway, the Irminger Sea and the Western half of the Labrador Sea, and secondarily, the Northern half of the North Sea, the Gulf of Lions and the Bay of Biscay. Conversely, in the areas where the primary production is weak (as in the subtropical gyre and in the Norwegian Sea), it remains weak, whatever the wind perturbations.

Furthermore, one particular ensemble member (Fig. 6.1, bottom left panel) may be well below the median in some regions (e.g. in the Labrador Sea) and

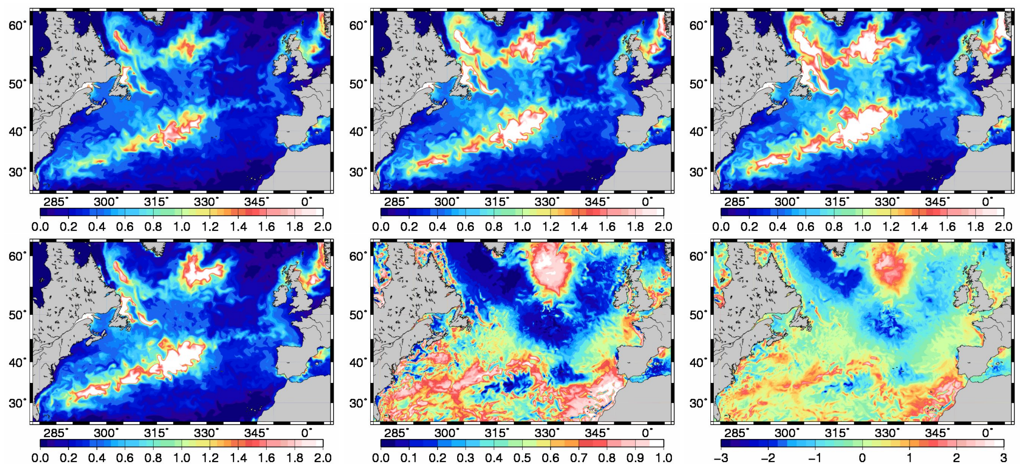

Figure 6.1: Deciles of the ensemble for phytoplankton (top panels) corresponding (from left ot right) to $r_k = 0.2$, 0.5 (median) and 0.8, and illustration of one of the ensemble members (bottom panels): the phytoplankton map (left panel), the rank in the ensemble (middle panel), and the transformed map (right panel) after Gaussian anamorphosis.

well above the median in other regions (e.g. in the Irminger Sea). This phenomenon is more obvious if we look at the rank of this ensemble member in the ensemble forecast (Fig. 6.1, bottom middle panel). More precisely, what is shown is the rank divided by the ensemble size (to be between 0 and 1), which corresponds to the local anamorphic transformation of the ensemble member using the uniform distribution $\mathcal{U}(0, 1)$ as a target distribution. For instance, a value below 0.2 means below the second decile ($r_2 = 0.2$), a value below 0.5 means below the median ($r_5 = 0.5$), etc. In this figure, we can see immediately where this ensemble member is high or low with respect to the others (compare the rank in the Labrador Sea and in the Irminger Sea), even in regions where the dispersion of the ensemble is very small, as along the coast of Africa or in the Southern half of the North Sea. See also how the high rank region in the Irminger Sea (i.e. with a production well above the ensemble median) embeds indifferently areas of high primary production and areas of low production, as a result of a strongly positive wind anomaly covering the whole region. The rank may thus better translate the effect of a homogeneous perturbation, which is masked in the original variable by the heterogeneity of the ecosystem dynamics. And from the local rank (Fig. 6.1, bottom middle panel) to the local Gaussian anamorphic transformation of the same ensemble member (Fig. 6.1, bottom right panel), there is nothing but a global anamorphosis transforming $\mathcal{U}(0, 1)$ into $\mathcal{N}(0, 1)$. The figure thus looks very similar, with the same nonlinear change of variable at every grid point (we could have kept the same figure, with a nonlinear labelling of the coluorbar).

This benchmark, and many other [Brankart et al., 2012, see] have also been used to evaluate the effect of local anamorphic transformations on the spatial correlation structure. As a general rule, the results indicate that

1. the transformation is accurate enough to faithfully preserve the correlation structure if the distribution is already close to Gaussian, and

2. the transformation has the general tendency of increasing the correlation radius as soon as the dependence between random variables becomes nonlinear.

These effects may be understood by observing that the linear correlation coefficient (Pearson) between the transformed variables corresponds to a nonlinear measure of correlation between the original variables, which is very similar to the rank correlation (Spearman). On the other hand, even if the method finds its full justification with a stochastic ensemble description of the uncertainties, several examples show that it may also be useful with the non-stochastic ensembles (resulting for instance from the system past variability) that are often used in present-day operational systems to reduce the numerical cost of data assimilation (until truly stochastic solutions become affordable). In both cases, the most important consequence for data assimilation of this increase in the correlation magnitude is a significant reduction in the number of degrees of freedom in the uncertainties (in a Gaussian sense), so that a better estimation accuracy can be obtained from a given observation network. And from a more general point of view, this also means that it may sometimes be rewarding to put some time and numerical effort to improve the statistical description of the uncertainties, rather than giving too much confidence to over simplistic assumptions.

Conclusions

The implementation in Matlab/Octave and/or Fortran languages, most compliant with specifications in work package 1, for each new nonlinear filter developed in SANGOMA has been described. The methods themselves are described in detail in Deliverable 3.5. We concentrated on the Equal Weights Particle filter or EWPF from UREAD, the Implicit Equal Weights Particle Filter or IEWPF from UREAD, the Nonlinear Ensemble Transform Filter or NETF from AWI and Multivariate Rank Histogram Filter from CNRS/LGGE.

The main message is that it is very easy to implement these new methods as several parts of these new schemes are similar to what is already available in the different toolboxes, and code can be reused. This provides another example of the efficiency of the toolboxes that have been developed further in SANGOMA. Another major advantage is that it is extremely easy to compare different methods and decide on the most efficient and accurate method for each specific application.

Bibliography

M. Ades and P. J. van Leeuwen. An exploration of the equivalent weights particle filter. *Quarterly Journal of Royal Meteorological Society*, pages n/a–n/a, 2012. doi: 10.1002/qj.1995.

J.-M. Brankart, C.-E. Testut, D. Bèal, M. Doron, C. Fontana, M. Meinvielle, P. Brasseur, and J. Verron. Towards an improved description of ocean uncertainties: effect of local anamorphic transformations on spatial correlations. *Ocean Science*, 8:121–142, 2012.

P.A. Browne and S. Wilson. A simple method for integrating a complex model into an ensemble data assimilation system using mpi. *Environmental Modelling & Software*, **68**:122–128, 2015.

E. Lorenz. Deterministic nonperiodic flow. *J. Atmos. Sci.*, 20:130–141, 1963.

E. Lorenz and K. Emmanuel. Optimal sites for supplementary weather observations. *J. Atmos. Sci.*, **55**:399–414, 1998.

S. Metref, E. Cosme, C. Snyder, and P. Brasseur. A non-gaussian analysis scheme using rank histograms for ensemble data assimilation. *Nonlin. Processes Geophys.*, 21:869–885, 2014.

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

Julian Tödter and Bodo Ahrens. A Second-Order Exact Ensemble Square Root Filter for Nonlinear Data Assimilation. *Mon. Wea. Rev.*, 143(4):1347–1367, 2015. ISSN 0027-0644.

Julian Tödter, Paul Kirchgessner, Lars Nerger, and Bodo Ahrens. Assessment of a nonlinear ensemble transform filter for high-dimensional data assimilation. *Mon. Wea. Rev.*, Under Revision, 2015.

P. J. Van Leeuwen. Nonlinear data assimilation in qeosciences: an extremely efficient particle filter. *Quarterly Journal of the Royal Meteorological Society*, **136**:1991–1999, 2010.

M. Zhu, P. J. van Leeuwen, and J. Amezcua. Implicit equal-weights particle filter. *Q. J. R. Meteorol. Soc.*, page personal communication, 2015.