Essay about Elitist Particle Filtering

Nadine Kuhnert

March 12, 2015

Contents

1 Elitist Particle Filtering

Algorithms like Particle Filters estimate the posterior probability distribution of a linear or non-linear state-space model based on Bayesian recursion equations. A particle filter is a so-called Sequential Monte Carlo (SMC) method, represents the posterior probability by a set of particles and generates samples from the required distribution.[Can11] Each sample is represented by weighted particles, where the weights represent the probability of the respective particle to be sampled from the true probability density function. There is no need to make assumptions on the structure of the state-space model or the probability distributions of the random variables.
In [HHM$^+$14], evolutionary strategies are combined with particle filtering in order to realize system identification with large search spaces. Elitist particles are selected based on long-term fitness measures and used to define new particles. Only the non-elitist particles are updated using the approximated PDF. In the following, the threshold of the weights to decide for a particle to be elitist is abbreviated by *elitThresh* and the number of elitist particles by *nElit*.
Two different models to describe the dynamic state-space equations are presented in the next subsections. The Univariate Nonstationary Growth Model (UNGM) is of interest in economics and the Bearings-Only Tracking model is popular in defense applications.

1.1 Univariate Nonstationary Growth Model

The Univariate Nonstationary Growth model described by Kotecha [KDM03] is highly nonlinear and bimodal in nature if $y_n > 0$. The dynamic state-space equations are defined as

$$x_n = \alpha x_{n-1} + \beta \frac{x_{n-1}}{1 + x_{n-1}^2} + \gamma \cos(1.2(n-1)) + u_n \qquad (1)$$

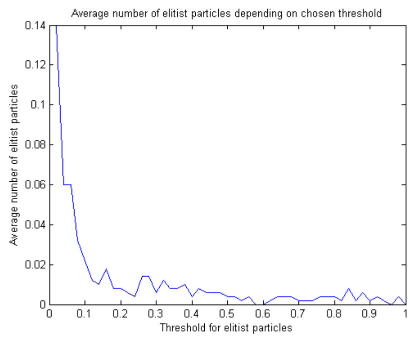
$$y_n = \frac{x_n^2}{20} + v_n \qquad n = 1, ..., N, \qquad (2)$$

using $x_0 = 0.1$, $\alpha = 0.5$, $\beta = 25$, $\gamma = 8$, $N = 500$ and
$v_n \sim \mathcal{N}(0, \sigma_v^2)$, $u_n \sim \mathcal{N}(0, \sigma_u^2)$, $\sigma_v^2 = 1$, $\sigma_v^2 = 1$.

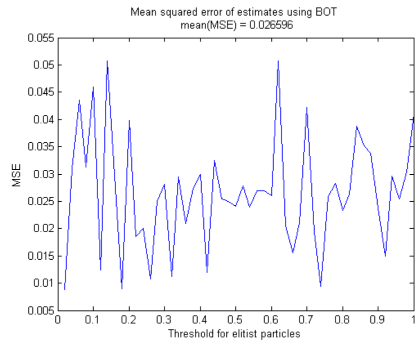
1.2 Bearings-Only Tracking Model

In this essay, we focus on the Bearings-Only Tracking model as it suits engineering problems better due to the fact that it is motivated by tracking a moving object in 2D.[KDM03] Thus, the process model is represented by a four-dimensional vector $\mathbf{x}_n$ where the x- and y- coordinates are denoted as x_n and y_n as well as the velocities in x- and y- direction are denoted as v_{xn} and v_{yn}, respectively:

$$\mathbf{x}_n = \mathbf{\Phi} \mathbf{x}_{n-1} + \mathbf{\Gamma} \mathbf{w}_n \qquad n = 1, ..., N, \qquad (3)$$



(a) Average number of elitist particles



(b) Mean squared error

Figure 1: Average number of elitist particles and the mean squared error drawn against the threshold for elitist particles

where $\mathbf{x}_n = [x_n, v_{xn}, y_n, v_{yn}]^T, \mathbf{w}_n = [w_{xn}, w_{yn}]^T$,

$$\mathbf{\Phi} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \mathbf{\Gamma} = \begin{bmatrix} 0.5 & 0 \\ 1 & 0 \\ 0 & 0.5 \\ 0 & 1 \end{bmatrix}.$$

The system noise is represented by $\mathbf{w}_n \sim \mathcal{N}(0, \sigma_w^2 \mathbf{I}_2)$ using $\mathbf{I}_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$ and $\sigma_w = 0.001$.
The measurements equation can be written as

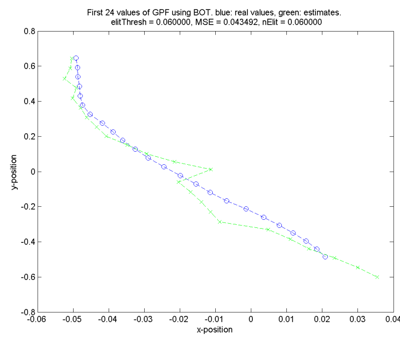
$$z_n = \tan^{-1}\left(\frac{y_n}{x_n}\right) + v_n \tag{4}$$

and stands for the measurements of a sensor corrupted by a Gaussian term v_n, where $v_n \sim \mathcal{N}(0, \sigma_v^2)$ using $\sigma_v = 0.005$. The initial state is estimated by $\mathbf{x}_n \sim \mathcal{N}(\mu_0, \mathbf{P_0})$ using $\mu_0 = [-0.05, 0.001, 0.7, -0.055]^T$ and

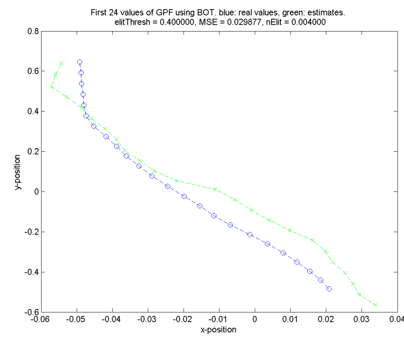
$$\mathbf{P_0} = \begin{bmatrix} 0.1 & 0 & 0 & 0 \\ 0 & 0.005 & 0 & 0 \\ 0 & 0 & 0.1 & 0 \\ 0 & 0 & 0 & 0.01 \end{bmatrix}.$$

All graphics are produced using $N = 500$ iterations and $M = 100$ samples. In the Figures 1(a) and 1(b), the dependence of the parameters on the chosen *elitThresh* is illustrated. The average number of elitist particles experiences a large decay with a growing *elitThresh* while the MSE is strongly fluctuating.

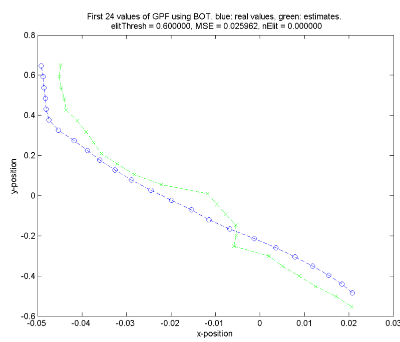
The trajectory of a moving object in 2D can be seen in Figure 2. The blue curves represent the true values, while the green curves stand for the estimated positions of the object. The estimated values vary with the chosen *elitThresh*.



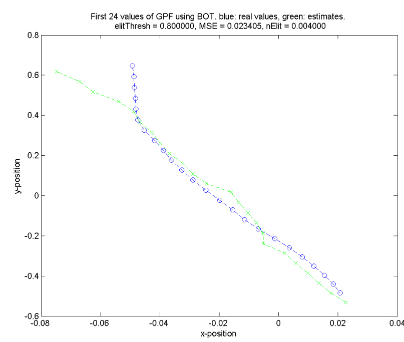
(a) elitThresh = 0.06



(b) elitThresh = 0.40



(c) elitThresh = 0.60



(d) elitThresh = 0.80

Figure 2: Tracking of a moving target. The true trajectory is drawn in blue, the estimated one in green for different *elitThresh*.

2 EPFES Using Gaussian-Weighted Weights

In order to measure the validity of the particles we computed the weights according to [KDM03]

$$w_n = \exp\left(-\frac{1}{2}\left(\frac{z - z_s}{\sigma_v}\right)^2\right), \tag{5}$$

where z represents the true value and z_s the estimated value of the measurement equation. w_n is normalized such that the weights sum up to one.

We modified this weighting by applying an additional Gaussian weighting of the weights. Our two approaches combine the EPFES and the GPF in two different ways. On the one hand, Gaussian weighting was applied only to the weights of the non-elitist particles. On the other hand, we weighted the weights of the elitist particles with a Gaussian function and computed the new estimates using only the elitist particles.

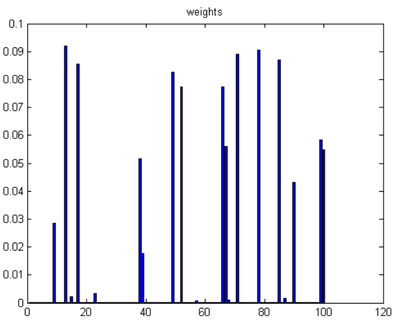


Figure 3: Weights of the particles

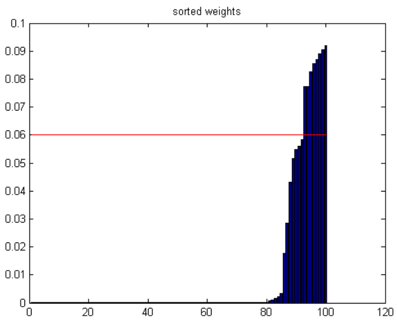


Figure 4: Sorted weights of the particles, the red line highlights the *elitThresh*

2.1 Gaussian Weighting Applied to Weights of Non-Elitist Particles

New estimates are computed using all particles while the weights of the non-elitist particles have been additionally weighted with a Gaussian function. The different steps of the algorithm are described and shown below. The following graphics are produced using the threshold for elitist particles $elitThresh = 0.06, M = 100$ and the samples after the first iteration $N = 1$.

1. Sort particles depending on their weight (ascending)

   ```
   w_nAndXmatrix = zeros(M,5);
   w_nAndXmatrix(:,1) = w_n;
   w_nAndXmatrix(:,2:5) = x_s;
   sortedMatrix = sortrows(w_nAndXmatrix);%sort matrix by weights (=first row)
   ```

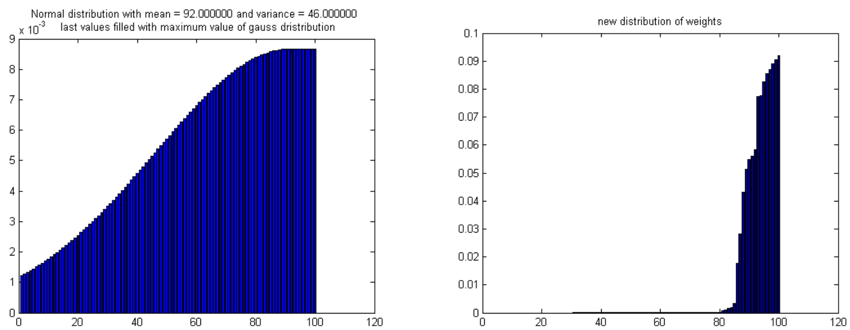
   The weights of all particles are drawn in Figure 3 compared to the Figure 4 showing the sorted weights and the later applied *elitThresh* highlighted in red.

2. Draw the Gaussian function from 1 to number of elitist particles (nEP) with mean equal to $M - nEP$, here $100 - 8 = 92$ and variance equal to half of mean, here $var = 46$ (see Figure 5(a))

3. Multiply the weights of the elitist particles with the computed Gaussian function (see Figure 5(b))

4. Compute new mean μ and variance σ^2 of the weighted particles $\hat{x}_n$

$$\mu = \sum_{i=1}^{M} \hat{x}_n(i) \qquad (6)$$



(a) Gaussian Function

(b) Weights Weighted with the Gaussian Function

Figure 5: Gaussian function with parameters depending on the *elitThresh* and the result of weighting the sorted weights with the Gaussian function

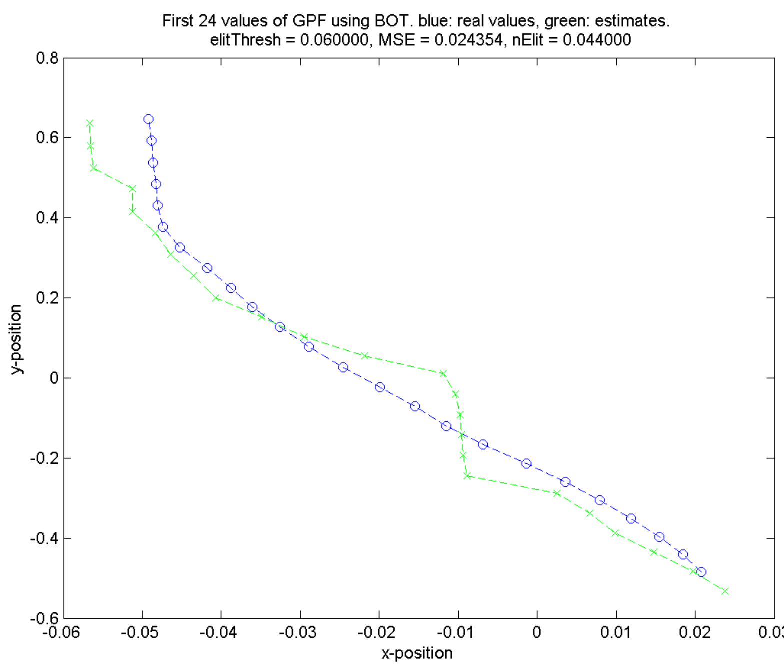
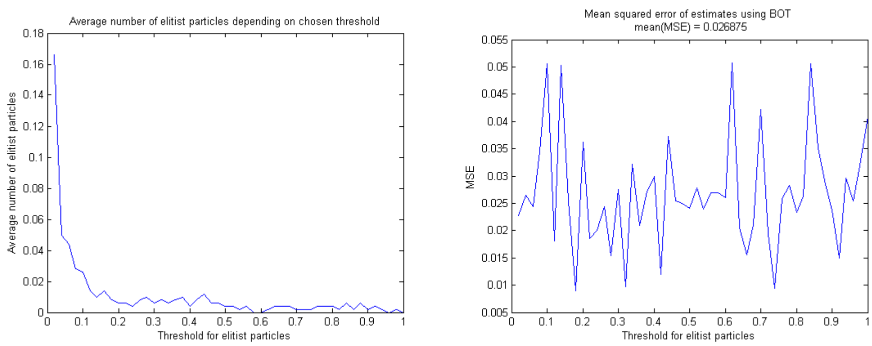


Figure 6: True (blue) and estimated (green) trajectory using elitThresh = 0.06

$$\sigma^2 = \sum_{i=1}^{M} (\hat{x}_n(i) - \mu)^2 \tag{7}$$



(a) Average number of elitist particles (b) Mean squared error

Figure 7: Average number of elitist particles and the mean squared error drawn against the threshold for elitist particles

5. Update all particles using the computed mean and variance

$$x_s(i) = \mu + rand \cdot \sigma, \qquad \text{for } i = 1, ..., M \tag{8}$$

   In Figure 6, the trajectory of the moving object can be seen. The blue curve illustrates the true positions in comparison to the estimated values in green using the *elitThresh* equal to 0.06 and the Gaussian weighting applied to the non-elitist particles.

6. The MSE (see Figure 7(b)) and the number of elitist particles (see Figure 7(a)) vary depending on the chosen *elitThresh*; Thus, we can evaluate the *elitThresh* be means of the MSE

2.2 Gaussian Weighting Applied to Weights of Elitist Particles

New estimates are computed using only elitist particles while the weights of the elitist particles have been additionally weighted with a Gaussian function. The different steps of the algorithm are described and shown below. The following graphics are produced using the threshold for elitist particles $elitThresh = 0.06, M = 100$ and the samples after the first iteration $N = 1$.

1. Sort particles depending on their weight (ascending)

```
w_nAndXmatrix = zeros(M,5);
w_nAndXmatrix(:,1) = w_n;
w_nAndXmatrix(:,2:5) = x_s;
sortedMatrix = sortrows(w_nAndXmatrix);%sort matrix by weights (=first row)
```

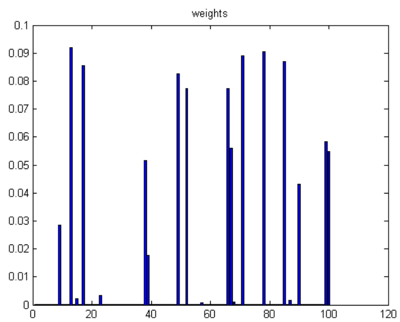


Figure 8: Weights of the particles

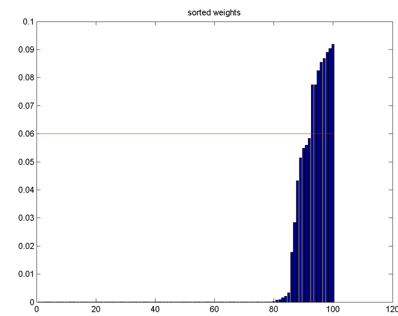
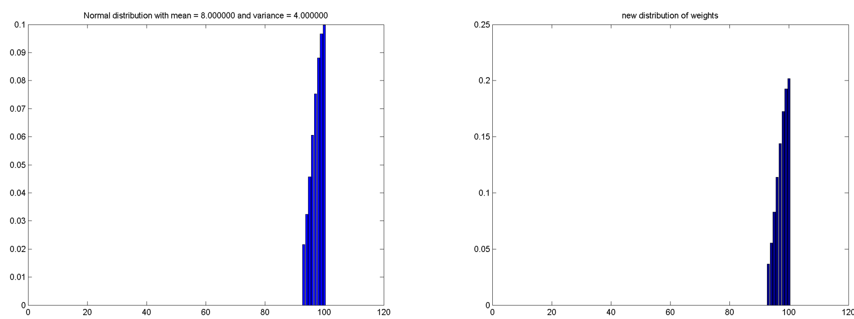


Figure 9: Sorted weights of the particles, the red line highlights the *elitThresh*



(a) Gaussian function

(b) Weights weighted with the Gaussian function

Figure 10: Gaussian function with parameters depending on the *elitThresh* and the result of weighting the sorted weights with the Gaussian function

The weights of all particles are drawn in Figure 8 compared to Figure 9 showing the sorted weights and the later applied *elitThresh* highlighted in red.

2. Draw Gaussian function from number of elitist particles (nEP) to M with mean equal to $nEP = 8$ and variance equal to half of nEP, here $var = 4$ (see Figure 10(a))

3. Multiply the weights of the elitist particles with the computed Gaussian function (see Figure 10(b))

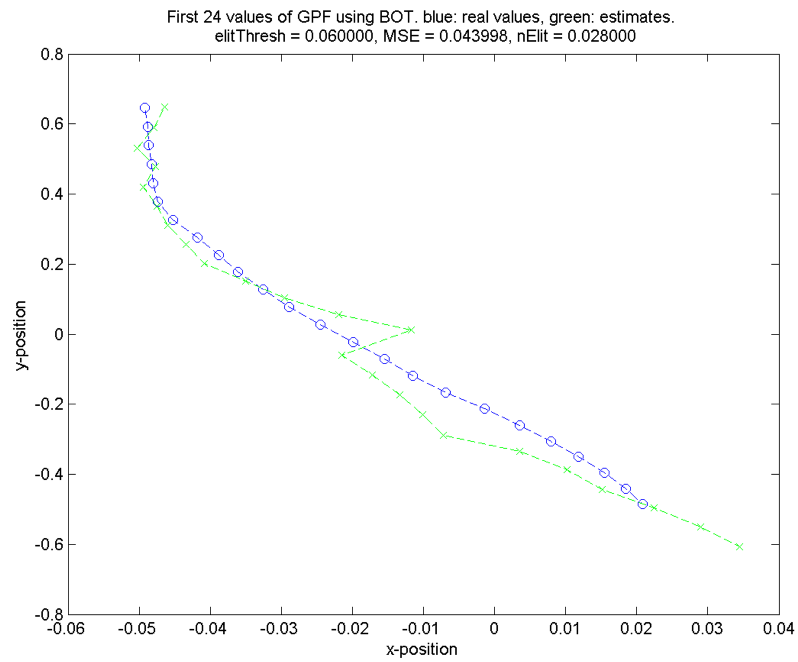


Figure 11: True (blue) and estimated (green) trajectory using elitThresh = 0.06

4. Compute new mean μ and variance σ^2 of the weighted elitist particles $\hat{x}_n$

$$\mu = \sum_{i=1}^{M} \hat{x}_n(i) \tag{9}$$

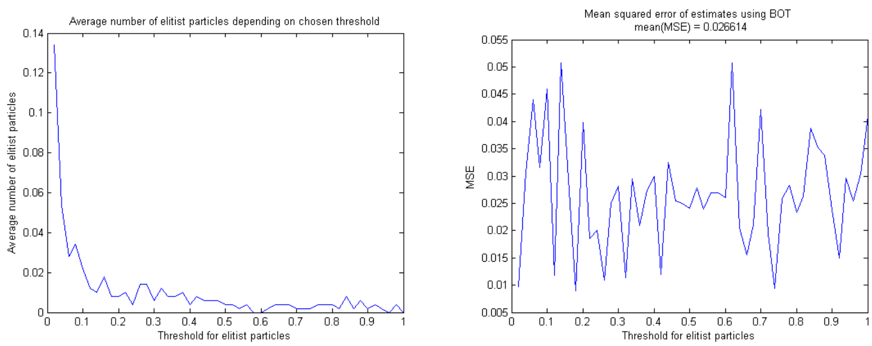
$$\sigma^2 = \sum_{i=1}^{M} (\hat{x}_n(i) - \mu)^2 \tag{10}$$

5. Update only the non-elitist particles using the computed mean μ and variance σ^2

$$x_s(i) = \mu + rand \cdot \sigma, \qquad \text{for } i = 1, ..., M - nElit \tag{11}$$

   In Figure 11, the trajectory of the moving object can be seen. The blue curve illustrates the true positions in comparison to the estimated values in green using the *elitThresh* equal to 0.06 and the Gaussian weighting applied to the elitist particles.

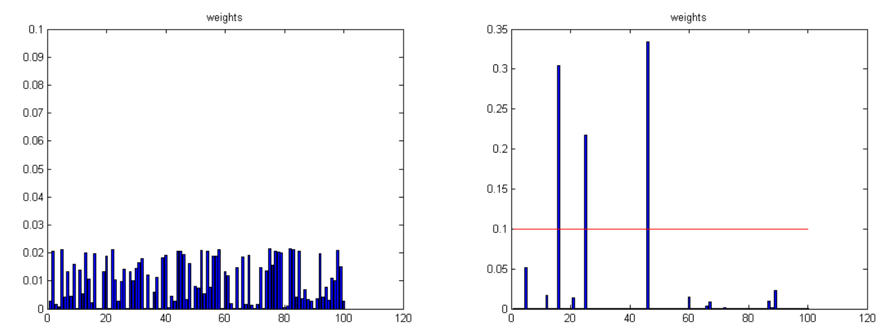
6. The MSE (see Figure 12(b)) and the number of elitist particles (see Figure 12(a)) vary depending on the chosen *elitThresh*; Thus, we can evaluate the *elitThresh* be means of the MSE.



(a) Average number of elitist particles

(b) Mean squared error

Figure 12: Average number of elitist particles and the mean squared error drawn against the threshold for elitist particles



(a) elitThresh =0.02, N=12

(b) elitThresh=0.06, N=13

Figure 13: Distribution of the weights using different values for *elitThresh*

3 Threshold for Elitist Particles Depending on the Weight Distribution

We saw previously, for example in Figure 12(b), that the results like the mean squared error are highly dependent on the chosen threshold for elitist particles. If all of the particles represent the true value in an approximately equal reasonable way, their weights will tend to a uniform distribution as shown in Figure 13(a). If only a few particles lie close to the true value, illustrated in Figure 13(b), the same threshold (drawn as a red line) will be more likely reached by these weights. Thus, we chose the threshold for elitist particles adaptively depending on the weight distribution. In the first case, we set the threshold according to half of the maximal value of the weights, in the second case, the threshold is set to the weights' variance. In order to receive a more reliable output, the results are based on the mean value of 50 realizations.

3.1 Threshold Equal to Half of Maximal Value of the Weights

The following graphics illustrate the development over 50 realizations of the MSE (see Figure 14(a)) and the number of elitist particles (see Figure 14(b)). The according threshold can be read from Figure 14(c) which is equal to half of the maximal value of the weights.

3.2 Threshold Equal to the Variance of the Weights

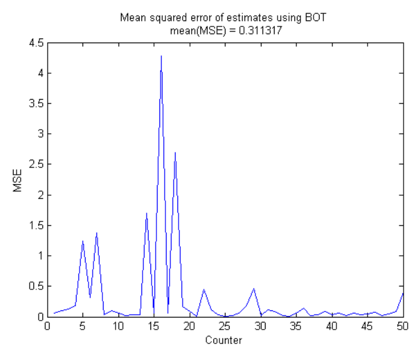
In this subsection, the development over 50 realizations of the MSE (see Figure 15(a)) and the number of elitist particles (see Figure 15(b)) is shown. The according threshold can be read from Figure 15(c) which is equal to the variance of the weights.

4 Summary

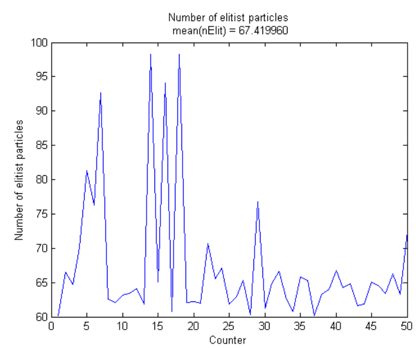
In the previous sections, we analysed elitist particle filtering based on evolutionary strategies with additional Gaussian weighting of the weights and adaptive thresholds for the elitist particles. The following abbreviations listed in Table 1 are used in the figures to address the different approaches. In order to compare the five methods, we computed the maximal, minimal and mean value of the mean squared error (MSE) illustrated in Figure 16.
We acheived the lowest minimal MSE represented by 0.0084 if we use the method *VarWeight*. However, this leads to the highest MSE (0.3189) in average. There was only a small difference between the mean MSE of the methods using additional Gaussian weighting. The second lowest average MSE (0.0269) provided *GaussOnElit*, while the lowest mean MSE (0.0266) can be reached by applying *GaussOnAll* or *NoGauss*. The methods *GaussOnAll*, *NoGauss* as well as *GaussOnElit* leaded to the lowest maximal MSE (0.0507), while we received even the highest minimal MSE (0.0090) with *GaussOnAll* and *GaussOnElit* compared to the other approaches. Evaluating the results of the MSE using *MaxWeight*, we observed the highest maximal value of MSE (4.2755), the second highest mean of MSE (0.3113) and the second lowest minimal value of MSE (0.0085). We obtain the minimal MSE of 0.0088 with *NoGauss* which is in between of the values of all methods.
In conclusion, we argue from the comparison of the MSE that on average the most reasonable positions can be computed using *GaussOnAll* or *NoGauss*. These methods lead to the lowest mean MSE (0.0266). However, if the threshold is chosen in the most suitable way, we can reach the best results with *VarWeight* obtaining the lowest minimal MSE (0.0084).

Table 1: Abbreviations of the compared approaches

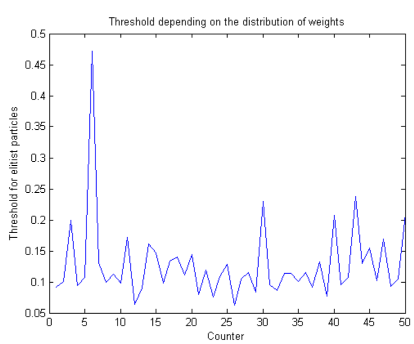
Abbreviation	Explanation
MaxWeight	Use an adaptive threshold equal to half of the maximal value of the weights while the weights are not additionally weighted
VarWeight	Use an adaptive threshold equal to the variance of the weights while the weights of the particles are not additionally weighted
GaussOnElit	Use a fixed threshold while the weights of the elitist particles are weighted with a Gaussian function and only the non-elitist particles are updated
GaussOnAll	Use a fixed threshold while the weights of the non-elitist particles are weighted with a Gaussian function and all of the particles are updated
NoGauss	Use a fixed thresshold while the weights of the particles are not additionally weighted



(a) Mean squared error

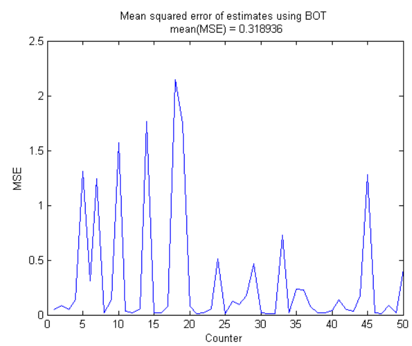


(b) Average number of elitist particles

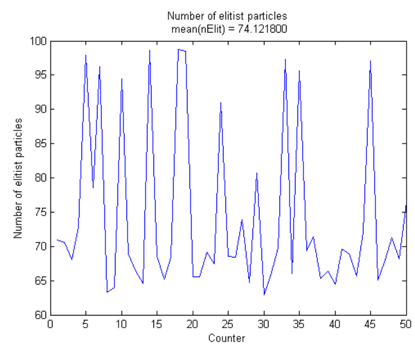


(c) elitThresh

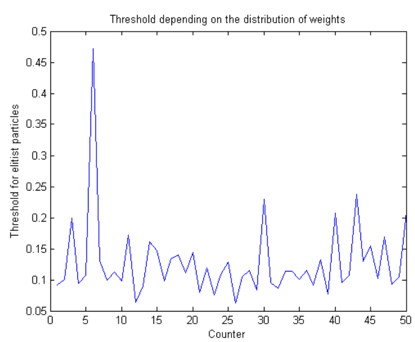
Figure 14: Results using an adaptive threshold equal to half of the maximal value of the weights



(a) Mean squared error

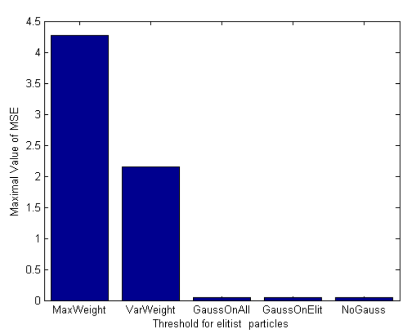


(b) Average number of elitist particles

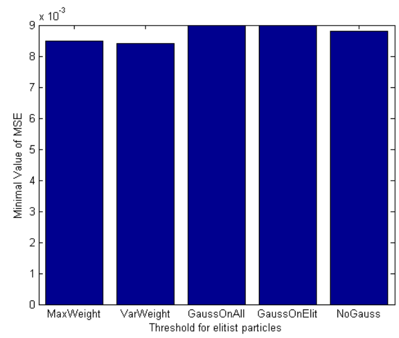


(c) Results using an adaptive threshold equal to the weights' variance

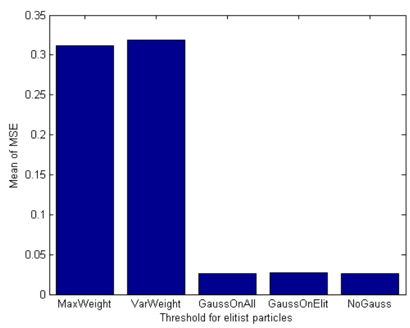
Figure 15: Results using an adaptive threshold equal to half of the maximal value of the weights



(a) Maximal value of MSE



(b) Minimal value of MSE



(c) Mean value of MSE

Figure 16: Comparison of the MSE of the different approaches

References

[Can11] Candy, J. V.: *Bayesian Signal Processing: Classical, Modern and Particle Filtering Methods.* Wiley, 2011 (Adaptive and Cognitive Dynamic Systems: Signal Processing, Learning, Communications and Control)

[HHM+14] Huemmer, C. ; Hofmann, C. ; Maas, R. ; Schwarz, A. ; Kellermann, W.: The elitist particle filter based on evolutionary strategies as novel approach for nonlinear acoustic echo cancellation. In: *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2014, Florence, Italy, May 4-9, 2014*, 2014, 1315–1319

[KDM03] Kotecha, J. H. ; Djuri, P. M. ; Member, Senior: Gaussian particle filtering. In: *IEEE Transactions on Signal Processing* 51 (2003), S. 2592–2601