18th International Conference on Information Fusion
Washington, DC - July 6-9, 2015

Improvements in the Implementation of Log-Homotopy Based Particle Flow Filters

Muhammad Altamash Khan, Martin Ulmke
Sensor Data and Information Fusion Department
FKIE Fraunhofer, Wachtberg, Germany
Email: altamash.khan@fkie.fraunhofer.de , martin.ulmke@fkie.fraunhofer.de

***Abstract*—State estimation of a non-linear system perturbed by non Gaussian noise is a challenging task. Typical solutions like EKF/UKF could fail while Monte Carlo methods, even though more accurate, are computationally expensive. Recently proposed log homotopy based particle flow filter, also known as Daum-Huang filter (DHF) provides an alternative way of non-linear state estimation. There have been a number of DHFs derived, based on solutions of the homotopy flow equation. The performance of these new filters depends a lot on the implementation methodology. In this paper, we highlight the key factors affecting the DHF performance and investigate them individually. We then make recommendations based on our results. It is shown that a properly designed DHF can outperform a basic particle filter, with less execution time.**

Keywords*—Particle flow filters , Log-homotopy , DHF , Multiple target tracking, Coupled model , Non-Gaussian noise, Stiff ODE, Shrinkage estimators, Redrawing, Kernel density estimation.

I. Introduction

The Bayesian estimation (BE) framework offers an intuitive way for the estimation of the hidden states of a dynamical system based on the observation data. The Bayesian estimation is carried out recursively, typically consisting of a prediction and a correction step. Finite dimensional analytical solutions to the BE problem are available only in few cases, mainly when the system model is linear Gaussian (Kalman filter) or a finite state Hidden Markov model (HMM) Traditional methods for non-linear state estimation include Extended (EKF) and Unscented Kalman filter (UKF). However these methods are usually sub-optimal and their performance degrades with the increase in the non-linearity, and also when the transition and measurement densities are non-Gaussian (e.g. multimodal, exponential).

Particle filters, also known as sequential Monte Carlo (SMC) methods, provide an alternative way to the state estimation. The main idea is to represent the posterior density by a weighted set of random samples (particles), which are then used to form the point estimates e.g. mean and variance [1]. Several version of particle filters have been proposed in the literature e.g. Sampling importance re-sampling (SIR) filter [2] also known as bootstrap particle filter, Auxiliary sampling importance resampling (ASIR) filter [3] , regularized particle filter (RPF) [4] etc. While particle filters can effectively deal with the non-linearities and non-gaussian noises, they suffer from the so called *weight degeneracy* and *curse of dimensionality*. A different approach to non-linear filtering has been suggested by Daum and Huang , which is based on the gradual inclusion of the measurements. The key idea is to model the transition of particles from the prior to the posterior density as a physical flow under the influence of an external force (measurements). Stochastic differential equations (SDE) define the flow of particles in pseudo-time, while the Foker-Plack equation (FPE) describes the density evolution. A flow vector is obtained by solving the FPE under different assumptions, which is then integrated numerically yielding updated particles states. The new filter is termed as homotopy based particle flow filter or simply Daum-Huang filter (DHF) after the developers. Different flow solutions have been derived, including the incompressible flow [5], zero diffusion exact flow [6], and non zero diffusion flow [7].

DHF implementations have been mentioned in several sources. While conceptually being quite intuitive, DHF performance suffers in practice due to several assumptions, made both in the theory and the implementation. In this paper we identify key factors/steps affecting the performance of the DHF, and suggest possible improvements in their implementation. We consider a non-linear and non-Gaussian estimation problem and show that by carefully choosing the methods for those key steps, DHF performance can be substantially improved over the more traditional implementations. We present a description of homotopy based particle flow in section II. In section III we describe our main contributions in detail. The specific dynamical model used in the study is outlined in the section IV. Simulation methodology and results are described in section V, which is followed by the discussion in section VI. Finally the conclusion is given in section VII.

II. Log homotopy based particle flow filters

Let $\mathbf{x}_k \in \mathbb{R}^d$ denote the state vector and $\mathbf{z}_k \in \mathbb{R}^m$ denote the measurement vector at time *k*. Also let $\mathbf{Z}_k$ denote the set of measurements up to time *k* including $\mathbf{z}_k$, $\mathbf{Z}_k = \{\mathbf{z}_1, \mathbf{z}_2 , ... , \mathbf{z}_k \}$. The state space model can be expressed in the terms of conditional probabilities,

$$\mathbf{x}_{k+1} \sim p(\mathbf{x}_{k+1}|\mathbf{x}_k) \quad (1)$$
$$\mathbf{z}_{k+1} \sim p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}) \quad (2)$$

$p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ and $p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})$ are referred to as the transition and the measurement(likelihood) densities. Assuming additive process and measurement noises w_k and v_k we can write

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k) = p_{w_k}(\mathbf{x}_{k+1} - \phi_k(\mathbf{x}_k)) \quad (3)$$
$$p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}) = p_{v_k}(\mathbf{z}_{k+1} - \psi_k(\mathbf{x}_{k+1})) \quad (4)$$

where ϕ_k is termed as the process / dynamical model and ψ_k as the measurement model. According to the Bayes theorem the prior density $p(\mathbf{x}_{k+1}|\mathbf{Z}_k)$ and the posterior density $p(\mathbf{x}_{k+1}|\mathbf{Z}_{k+1})$ are recursively defined as,

$$p(\mathbf{x}_{k+1}|\mathbf{Z}_k) = \int p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{Z}_k)d\mathbf{x}_k \quad (5)$$

$$p(\mathbf{x}_{k+1}|\mathbf{Z}_{k+1}) = \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{Z}_k)}{p(\mathbf{z}_{k+1}|\mathbf{Z}_k)} \quad (6)$$

where $p(\mathbf{x}_k|\mathbf{Z}_k)$ is posterior density at time k. The conditional density $p(\mathbf{z}_{k+1}|\mathbf{Z}_k)$ appears as a normalization constant in the measurement update equation (6). The DHF as described in [5]-[7], shares the importance sampling step with the particle filter but it specifically uses the prior distribution of the state vector $p(\mathbf{x}_{k+1}|\mathbf{x}_k)$ as the importance density. The main difference lies in how the measurements are incorporated to derive the posterior density. The idea here is to model the motion of particles from the prior to the posterior densities, in a way analogous to the flow of physical particles. A log-homotopy function $\log p(\mathbf{x}_k, \lambda)$ is defined through the homotopy parameter λ,

$$\log p(\mathbf{x}_{k+1}, \lambda) = \log g(\mathbf{x}_{k+1}) + \lambda \log h(\mathbf{x}_{k+1}) - \log K(\lambda). \quad (7)$$

where $g(\mathbf{x}_{k+1})$ represents the prior $p(\mathbf{x}_{k+1}|\mathbf{Z}_k)$, $h(\mathbf{x}_{k+1})$ the likelihood $p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})$ and λ the artificial/synthetic time varying from 0 to 1. $K(\lambda)$ is the normalization constant for the posterior density independent of $\mathbf{x}_{k+1}$. λ = 0 sets $p(\mathbf{x}_{k+1}, \lambda)$ equal to the prior density while with λ = 1 the transformation is completed to the normalized posterior density. From now onwards, we drop the time index k for the sake of convenience and ignore the normalization constant $K(\lambda)$. It is supposed that the flow of particle obeys the Ito SDE,

$$d\mathbf{x} = f(\mathbf{x}, \lambda)d\lambda + \sigma(\mathbf{x}, \lambda)d\mathbf{w} \quad (8)$$

where $f(\mathbf{x}, \lambda)$ is the flow vector, $\mathbf{w}$ is the M-dimensional Wiener process with diffusion term $\sigma(\mathbf{x}, \lambda)$. State $\mathbf{x}$ is assumed to be an implicit function of λ. For a flow characterized as in (8), the evolution of the density $p(\mathbf{x}, \lambda)$ w.r.t the parameter λ is given by the Fokker-Planck equation (also known as Kolmogorov forward equation),

$$\frac{\partial p(\mathbf{x}, \lambda)}{\partial \lambda} = -\sum_{i=1}^{d} \frac{\partial}{\partial \mathbf{x}_i}[f_i(\mathbf{x}, \lambda)p(\mathbf{x}, \lambda)] + \frac{1}{2}\sum_{i=1}^{d}\sum_{j=1}^{d} \frac{\partial^2}{\partial \mathbf{x}_i \partial \mathbf{x}_j}[\mathbf{Q}_{i,j}(\mathbf{x}, \lambda)p(\mathbf{x}, \lambda)] \quad (9)$$

where $\mathbf{Q}_{i,j}(\mathbf{x}, \lambda)$ is the diffusion matrix. Different flow solutions have been obtained by solving equation (9) under different assumptions. Here we discuss the latest of them, which is termed as the non zero diffusion flow. The derivation of this flow is given in [7]. Here we only show the end result,

$$f(\mathbf{x}, \lambda) = \Big[\frac{\partial^2}{\partial \mathbf{x}^2} \log p(\mathbf{x}, \lambda)\Big]^{-1} \Big[\frac{\partial}{\partial \mathbf{x}} \log h(\mathbf{x})\Big]^T \quad (10)$$

with the constraint

$$\frac{\partial}{\partial \mathbf{x}}\Big[\frac{\partial}{\partial \mathbf{x}} \cdot f(\mathbf{x}, \lambda)\Big] + \Big[\frac{\partial}{\partial \mathbf{x}} \log p(\mathbf{x}, \lambda)\Big]\Big[\frac{\partial}{\partial \mathbf{x}} f(\mathbf{x}, \lambda)\Big]^T = \frac{\partial}{\partial \mathbf{x}}\left[\frac{1}{2p(\mathbf{x}, \lambda)} \frac{\partial}{\partial \mathbf{x}} \cdot (\mathbf{Q}(\mathbf{x}, \lambda) \cdot \frac{\partial}{\partial \mathbf{x}} p(\mathbf{x}, \lambda))\right] \quad (11)$$

The hessian of $\log p(\mathbf{x}, \lambda)$ can be approximated as

$$\frac{\partial^2}{\partial \mathbf{x}^2} \log p(\mathbf{x}, \lambda) = \frac{\partial^2}{\partial \mathbf{x}^2} \log g(\mathbf{x}) + \lambda \frac{\partial^2}{\partial \mathbf{x}^2} \log h(\mathbf{x}) \quad (12)$$

$$\approx -\mathbf{P}^{-1} + \lambda \frac{\partial^2}{\partial \mathbf{x}^2} \log h(\mathbf{x}) \quad (13)$$

where $\mathbf{P}^{-1}$ is the prior covariance matrix estimate. The hessian of the log-likelihood, $\frac{\partial^2}{\partial \mathbf{x}^2} \log h(\mathbf{x})$, can be calculated analytically in most cases.

III. Main Contribution

Numerical results for the DHF have been presented in [8]. DHF based on the incompressible and exact flows have been implemented by Choi. et.al. in [9] for non-linear scalar and linear vector system models. Exact flow DHF implementations for multi-target tracking have been reported in [10]. Also in [11], joint probabilistic data association (JPDA) and maximum aposteriori penalty function (MAP-PF) algorithms based on the exact flow DHF have been derived. While particle flow filters are theoretically quite elegant, their performance suffer from approximations made, both in theory and in the practical implementation. This includes approximations made while deriving the flow, estimation of the prior density and the use of numerical techniques. These lead to the introduction of bias and loss of asymptotic consistency [12].

There could be several ways in which a DHF can be implemented. Below, we outline the method described in [10].

```
Initialize DHF: Generate initial set of particles;
Initialize EKF/UKF: Initial mean and covariance;
Pseudo-time grid discretization;
for Loop over the time do
    Propagate particles using the dynamic model;
    Time update for EKF/UKF;
    Prior covariance matrix estimate from EKF/UKF;
    for Loop over the pseudo-time do
        for Loop over individual particles do
            Integration of the flow equation;
        end
        Measurement Update for EKF/UKF ;
        Mean state evaluation from particles;
        Redraw particles (Optional);
    end
end
```

Algorithm 1: Generic implementation of DHF

Particles are generated by sampling the transition density. An EKF/UKF is run in parallel to the main algorithm. This is done in order to approximate the prior covariance matrix. Next the flow equation is solved in the pseudo-time for all particles. The flow equation uses the prior covariance estimate from the parallel running EKF/UKF. Once done, the mean state vector is estimated and the measurement update is carried out for EKF/UKF. This process in repeated till the end of the simulation time.

The steps colored in red are crucial factors in the performance of the DHF. First comes the pseudo-time λ discretization strategy, together with the numerical integration method. As the DHF flow is described by an ordinary differential

equation (ODE), a suitable discretization is essential to capture the flow dynamics. Then the flow equation is integrated w.r.t. λ. While the exact implementational details for the references ([7], [9],[11]) are not known, authors in [10] have used single step Euler integration as mentioned in the pseudo-code. The method is based on first order Taylor series expansion. It is simple to implement and is fairly quick. But care has to be taken as the flow ODE can exhibit stiffness. In that case a straight forward λ discretization together with single step Euler integration might not work. Secondly, the non zero diffusion flow requires an estimate of the prior covariance matrix. While prior covariance estimate from parallel running EKF/UKF can be used as an approximation, this makes the DHF accuracy dependant on EKF/UKF. On the other hand, a sample covariance estimate can often be ill-conditioned. The question then becomes, is there a better method to estimate the prior covariance matrix. Finally, the re-generation of a new set of particles is an important step. Unlike a standard particle filter, the re-sampling/re-drawing step is not mandatory in the DHF but optional. Instead, it is descibed that the homotopy flow moves the particles to their correct location in the state space. But due to approximations, the flow might not be accurate. Hence the effect of particle re-generation is worth investigating. In the current work we look for improvements in the DHF performance by considering changes in the existing implementation architecture, as mentioned in Algorithm 1. We study the four factors mentioned in the last section, individually.

A. Pseudo-time discretization

While comparing the two flows, it was shown in [13] that the non zero diffusion flow is considerably stiff as compared to the exact flow, where authors used 39 exponentially spaced λ points for solving the ODEs. In this paper we consider both uniform and non-uniform grid discretization. The idea is to analyze the effect of a particular numerical integration scheme and grid discretization strategy on the filter performance, in terms of the estimation error and the processing time.

B. ODE numerical solution

The homotopy flow is defined by a vector ordinary differential equation (ODE). In the current work, we seek for the numerical solution of the ODE. Broadly speaking, ODEs can be catagorized as being stiff and non-stiff. While there is no precise definition of the stiffness, in the literature two criteria are generally mentioned for describing a stiff ODE. First, the condition number of the jacobian matrix $J(x, \lambda) = \frac{\partial f(x,\lambda)}{\partial x}$ of a stiff ODE is quite large. As a consequence, multiple timescales exist in the ODE. Time scales, often referred as modes, are defined by the inverse of the absolute of the eigenvalues. Secondly, in the Lipschitz's inequality $||f(x_2, \lambda_2) - f(x_1, \lambda_1)|| \leq L||\lambda_2 - \lambda_1||$, the Lipschitz's constant L is typically very high for a stiff ODE. Non-zero diffusion ODE can be characterized as stiff according to the both criteria. Therefore, care has to be taken while chosing the numerical integration scheme for solving the flow ODE.

The standard Euler's method is used for solving the flow ODE in earlier works. It is a first order method with the truncation error in the order of $\mathcal{O}(h^2)$. In this paper, we intend to compare the performance of some other numerical integration (NI) schemes for solving stiff ODEs alongside the Euler's method. There are several choices available. Below, we mention some of the common NI methods for solving stiff ODEs.

1) Forth order Runge-Kutta method: Forth order Runge-Kutta method (RK4) is our second integration method. The RK4 method is a fourth-order method, which implies that the local truncation error is on the order of $\mathcal{O}(h^5)$,while the total accumulated error is of order $\mathcal{O}(h^4)$.

2) Rosenbrock method: Rosenbrock methods are family of multistep procedures to solve stiff ODEs. Formulas use the Jacobian matrix into the integration formula. Like the Runge-Kutta methods, Rosenbrock methods successively form intermediate results. Therefore, they are also called Runge–Kutta–Rosenbrock methods.

3) Gear's method: The Gear's method belongs to the class of methods known as backward differentiation formulae (BDF). It is an implicit integration method and uses the first and higher order derivatives. It is a predictor-corrector type scheme.

C. Prior covariance shrinkage estimation

The evaluation of the flow equation (10) require the availability of the prior covariance estimate. This can be derived in several ways. The simplest way is to estimate the covariance matrix using the prior particles. This is referred to as the sample covariance estimate S. S is an unbiased estimator of the true prior covariance P, and is the maximum likelihood estimate if the data is Gaussian distributed. But in the presence of non-linear models, the Gaussian assumption does not valid. Also S could progressively get ill-conditioned. i.e. the spread of the eigenvalues gets larger with the passage of time. This is especially the case, when the $\frac{d}{N_p}$ ratio is non-negligable, where d is the state vector dimension and N_p is the number of particles. As a consequence, the matrix inversion could lead to stability issues. An alternative method suggested by authors in [14] is to run an EKF/UKF in parallel to DHF, and to use the prior covariance matrix generated by those filters. We refer to such matrix as P_{XKF}, where XKF could be Extended or Unscented version of the KF. While this method is better than using the raw data based covariance estimate, it ties the DHF estimation accuracy to that of the EKF/UKF. P_{XKF} could also exhibits a wide spread of the eigenvalues.

Therefore, we look for some other method of covariance matrix estimation. That method should have two properties. First, the resulting matrix should always be positive definite (PD) and second, the matrix should be well-conditioned [15]. *Shrinkage estimation* is an alternative method, used in the multivariate statistics literature for the estimation of the covariance matrix. The use of such methods date back to work of Stein [16]. The main idea is to merge the raw estimate (S) which is unbiased but normally with high variance, together with a more structured but typically a biased target (B) through a scale factor, to get the combined estimate (P_*). The objective is to reduce the estimation error, typically in mean squared sense (MSE), by achieving an optimal trade off between the biased (B) and the unbiased (S) estimators. The scale factor is also called shrinkage intensity ρ as it shrinks the eigenvalues of S optimally towards the mean of eigenvalues of the true

covariance matrix P. The resulting covariance matrix (P_*) will be biased, but will improve on the two aforementioned properties, and is hoped to lower the estimation error. There are several shrinkage estimators mentioned in the literature, with different target covariance matrices. In the current work, we describe some of the more established shrinkage estimators.

1) Shrinkage towards the Identity matrix: In subsections III-C1 to III-C3, shrinkage estimator is defined by convex combination of the matrices B and S. The objective is to find an optimal shrinkage intensity that minimizes the the MSE,

$$\min_{\rho} \quad E[||P_* - P||^2] \tag{14}$$

where $P_* = \rho B + (1-\rho)S$. Shrinkage towards the Identity matrix is described in [17]. The two main objectives defined are, to get an asymptotically consistent estimator that is more accurate than the sample covariance matrix S, and is well-conditioned. No prior structure is assumed about the target matrix B, as it could lead to an increased biasness. Instead a simple matrix with same covariance terms and zero cross-variances (scaled Identity) is chosen as the target. Hence, $B = m_n I$ with $m_n = tr(S)/d$. Also the shrinkage intensity is defined as $\rho = \frac{\hat{\alpha_1}}{\hat{\alpha_2}}$ where $\hat{\alpha_2} = ||S - m_n I||^2$ and $\hat{\alpha_1} = \hat{\alpha_2} - \min(\hat{\alpha_2}, \frac{1}{n^2}\sum_{i=1}^{N_p}[||X_n^i (X_n^i)^T - S||^2])$ $||.||$ is the Frobenius norm and X_n^i is the *ith* particle. One main advantage of P_{LW0} is that it does not assume any particular distribution for the data and therefore is *distribution-free*.

2) Shrinkage towards the constant correlation matrix: This estimator is derived in [18], in the context of portfolio optimization. The target matrix is chosen according to the constant correlation model. It means that pairwise correlations are identical, which is given by the average of all the sample correlations. We denote this estimator by P_{LW1}. The target matrix B is given by

$$B = \begin{cases} S_{ii} & : i = j \\ \bar{r}\sqrt{S_{ii}S_{jj}} & : i \neq j \end{cases}$$

where $\bar{r}$ is the average sample correlation. The shrinkage intensity is defined as $\rho = \max\{0, \min\{1, \frac{\kappa}{d}\}\}$, with the κ given as , $\kappa = \frac{\hat{\pi} - \hat{\varrho}}{\hat{\gamma}}$. $\hat{\pi}$ denotes the sum of asymptotic variances of the entries of the sample covariance matrix S, while $\hat{\varrho}$ denotes the sum of asymptotic covariances of the entries of the shrinkage target B with the entries of the sample covariance matrix. $\hat{\gamma}$ gives a measure of the misspecification of the shrinkage target. Formulas for $\hat{\pi}$, $\hat{\varrho}$ and $\hat{\gamma}$ are bit long and we do not show them here. Interested readers may consult the source article.

3) Shrinkage towards the perfect postive correlation matrix: Authors in [19] suggest single-factor matrix as the shrinkage target. The paper is concerned with estimating the structure of the risk in the stock market and the modelling of the stock returns. The fact that stock returns are positively correlated to each other, is exploited. The shrinkage target is given by,

$$B_{ij} = \begin{cases} S_{ii} & : i = j \\ \sqrt{S_{ii}S_{jj}} & : i \neq j \end{cases}$$

The resulting linear estimator is denoted as P_{LW2}. The shrinkage intensity has the same form as in for P_{LW1}, but with slighty different formula for $\hat{\varrho}$.

4) Emprical Bayesian: In [20], an estimator for multivariate Gaussian data is derived. It is given by the linear combination of the sample covariance matrix S and scaled identity matrix. The scaling factor is estimated from the data. We denote this estimator by P_{EB} and it is given by

$$P_{EB} = \frac{N_p d - 2N_p - 2}{N_p^2 d}[det(S)]^{\frac{1}{d}} I + \frac{N_p}{N_p + 1} S \tag{15}$$

5) Stein Haff: This estimator is described in [21]. The general form of the estimtor is $V(S)\Phi(l(S))V(S)^T$, where $V(S)$ matrix contains the eigenvectors of the sample covariance matrix S while $\Phi(l(S))$ is a matrix that is a fucntion of the eigenvalues $l(S)$ of the S. The data is assumed to be normally distributed. The Stein-Haff estimator denoted by P_{SH}, is contructed by leaving the eigenvectors of the S unchanged while replacing the eigenvalues l by $\tilde{\tilde{l}}_i = nl_i / \Big(n - p + 1 + 2l_i \sum_{j=1, j\neq i}^{N_p} \frac{1}{l_i - l_j}\Big)$. Eigenvalues can get disordered by the transformation and might become negative, which could lead to the covariance estimate lossing its positve definiteness. Therefore another algorithm called isotonic regression is used in conjunction with the transformation [22].

6) Minimax: The final shrinkage estimator considered is derived in [23]. Again Gaussian assumption is made. This estimator is termed minimax because under certain loss function, it has the lowest worst case error [17]. Its structure is similar to the P_{SH} but sample eigen values are replaced by $\tilde{l}_i = \frac{n}{n+p-1-2i} l_i$. This estimator is denoted here by P_{MX}. Isotonizing regression is not applied in this case.

D. Re-generating the particles set

In the standard particle filter, new set of particles are generated after the measurement inclusion step. This is done in order to avoid the particle degeneracy. A measure of the particle degeneracy is the effective number of particles N_{eff}.When N_{eff} falls below a certain threshold, resampling of the particles is carried out. Depending on the number of particles, this can be computationally expensive. Homotopy based particle flow filters try to avoid the particle degeneracy by the gradual inclusion of the measurements. Unlike standard particle filters, resampling is not a mandatory step in the DHF according to [14], as it moves the particles to the correct region of the state-space. However due to the inexactness of the homotopy flow ODE, the particle state update itself is imperfect. Hence the generation of a new particle set could potentially help in relocating/confining the particles to the correct region. Instead of the conventional resampling, an optional redrawing of the particles is hinted out by the Daum and Huang in their papers. We find a single source describing the particles redrawing method. In [10], it is suggested to redraw a new set of posterior particles by sampling a Gaussian distribution. The mean of the distribution is estimated using particles, while the filtered covariance matrix is provided by the EKF. However, the effect of the redrawing on the performace of DHF is not yet studied upto our knowledge. Therefore, our last contribution is to define a particle redrawing strategy, and to study the effect of different redrawing schemes on the performance of DHF. We consider two redrawing schemes, first using a single multivariate Gaussian distribution (MVG), and second using a

$$
\begin{aligned}
x^i_{k+1} &= x^i_k + \dot{x}^i_k \Delta t + \frac{1}{2} a_{x_{k+1}} \Delta t^2 \\
y^i_{k+1} &= y^i_k + \dot{y}^i_k \Delta t + \frac{1}{2} a_{y_{k+1}} \Delta t^2 \\
\dot{x}^i_{k+1} &= \dot{x}^i_k + \Pi^i_{x_k} \Delta t + a_{x_{k+1}} \Delta t \\
\dot{y}^i_{k+1} &= \dot{y}^i_k + \Pi^i_{y_k} \Delta t + a_{y_{k+1}} \Delta t
\end{aligned}
\qquad
\begin{aligned}
\Pi^1_{x_k} &= \frac{1}{N-1} \sum_{i=2}^{N} \Big(\frac{\kappa_1}{\sqrt{(x^1_k - x^i_k)^2 + (y^1_k - y^i_k)^2 + \delta}} \Big) \frac{v_t^2}{r_t} \cos(\frac{v_t}{r_t} k) \\
\Pi^1_{y_k} &= -\frac{1}{N-1} \sum_{i=2}^{N} \Big(\frac{\kappa_1}{\sqrt{(x^1_k - x^i_k)^2 + (y^1_k - y^i_k)^2 + \delta}} \Big) \frac{v_t^2}{r_t} \sin(\frac{v_t}{r_t} k) \\
\Pi^i_{x_k} &= \kappa_2 (x^1_k - x^i_k) - \kappa_3 \dot{x}^i_k \\
\Pi^i_{y_k} &= \kappa_2 (y^1_k - y^i_k) - \kappa_3 \dot{y}^i_k
\end{aligned}
\tag{D1}
$$

$$
r^i_{k+1} = \sqrt{(x^{(i)}_{k+1})^2 + (y^{(i)}_{k+1})^2} + v^i_{r_{k+1}}
\qquad
\theta^i_{k+1} = \tan^{-1} \Big(\frac{y^{(i)}_{k+1}}{x^{(i)}_{k+1}} \Big) + v^i_{\theta_{k+1}}
\tag{D2}
$$

$$
\begin{aligned}
p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}) &= p(\mathbf{r}_{k+1}|\mathbf{x}_{k+1}) p(\theta_{k+1}|\mathbf{x}_{k+1}) \\
&= \frac{1}{\Big(2\pi\beta^2\Big)^{\frac{N}{2}} |R_r|^{\frac{1}{2}}} \exp\Big\{ -\frac{1}{2} \big(\mathbf{r}_{k+1} - \tilde{\mathbf{r}}_{k+1}\big)^T R_r^{-1} \big(\mathbf{r}_{k+1} - \tilde{\mathbf{r}}_{k+1}\big) \Big\} \prod_{i=1}^{N} \exp\Big\{ -\frac{1}{\beta} \Big(\theta^{(i)}_{k+1} - \tan^{-1}(\frac{y^{(i)}_{k+1}}{x^{(i)}_{k+1}}) \Big) \Big\}
\end{aligned}
\tag{D3}
$$

$$
\tilde{\mathbf{r}}_{k+1} = \Big[\sqrt{(x^{(1)}_{k+1})^2 + (y^{(1)}_{k+1})^2} \;\; \sqrt{(x^{(2)}_{k+1})^2 + (y^{(2)}_{k+1})^2} \cdots \sqrt{(x^{(N)}_{k+1})^2 + (y^{(N)}_{k+1})^2} \Big]^T
\qquad
R_r = \begin{bmatrix} \sigma_r^2 & \sigma_{r_x}^2 & \cdots & \sigma_{r_x}^2 \\ \sigma_{r_x}^2 & \sigma_r^2 & \cdots & \sigma_{r_x}^2 \\ \vdots & \vdots & \vdots & \vdots \\ \sigma_{r_x}^2 & \sigma_{r_x}^2 & \cdots & \sigma_r^2 \end{bmatrix}
$$

Gaussian mixture model (GMM) that is estimated through the kernel density estimation (KDE).

1) Redrawing from MVG: Our first technique is similar to the one described in the pseudo-code in [10]. New particles are generated from a multivariate gaussian distribution. The mean of the distribution is estimated using the posterior DHF particles while shrinkage estimation is used to get the covariance matrix, as opposed to the [10] where posterior estimate from EKF is used.

2) Redrawing from KDE-GMM: KDE is a non-paramteric method to estimate the probability density of random variables. In this paper we use the online KDE approach described in Kristan et.al. [24], where two main contributions are made. First, the KDE of the target distribution is constructed by online clustering of the data points. Secondly, each new observation is treated as a distribution in the form of Dirac delta functions, and the sample distribution is modelled by a mixture of Gaussian and Dirac delta functions. Sample distribution is continously refined and compressed for keeping the algorithm complexity low.

The question, when to redraw a new particle set, is an important one. While redrawing can be carried out at each time step, it might not be very efficient. Instead it should be carried out when the particle spread gets too large.

$T_{th} = \tilde{a} + \exp(-\tilde{b}\tau)$;
$\Psi(k) = trP_{prior}(k) - trP_{post}(k)$;
if $\frac{\Psi(k)}{\Psi(k-1)} \geq T_{th}$ **then**
    Redraw;
    $\tau = 0$;
else
    No redraw;
    $\tau \leftarrow \tau + 1$;
end

Algorithm 2: Particle redrawing criterion

An indicator of the particle spread is the difference of the traces of the prior and the posterior covariance matrices Ψ. The idea is to observe the trend of the particle spread and redraw a new set of particles when the ratio $\frac{\Psi(k)}{\Psi(k-1)}$ gets above a certain redrawing threshold T_{th}, which is also time dependant. The reason for having a time dependant threshold T_{th} is that, one would like to wait for a certain time before a new redraw. This is accomplished here by reducing the T_{th} exponentially from the maximum value until the redrawing criteria is met. At that time, the T_{th} is reset and the process continues. The exponential decrease is controlled by the parameters $\tilde{a}$ and $\tilde{b}$, and the redrawing time index τ. Redrawing is explained in the Algorithm 2.

IV. Model description

Here, we consider the scenario similar to the one described in [13], namely the tracking of multiple targets in a 2D space using range and bearing measurements. States of targets are interdependant, therefore resulting in a non-linear coupled dynamical model. Furthermore, target association is assumed to be perfectly known and hence we do not use any data association algorithm. The state vector for the target *i* at time instant *k* is $\mathbf{x}^{(i)}_k = (x^{(i)}_k, y^{(i)}_k, \dot{x}^{(i)}_k, \dot{y}^{(i)}_k)$, where $x^{(i)}_k$ and $y^{(i)}_k$ represent the position while $\dot{x}^{(i)}_k$ and $\dot{y}^{(i)}_k$ representing velocity components along the x and y-axis respectively. The overall state vector is formed by concatenating the individual target state vectors $\mathbf{x}_k = [\mathbf{x}^{(1)}_k, \mathbf{x}^{(2)}_k \ldots \mathbf{x}^{(N)}_k]$. Also the measurement vector for the target *i* is given by $\mathbf{z}^{(i)}_k = (r^{(i)}_k, \theta^{(i)}_k)$, where $r^{(i)}_k$ is the range to the target while $\theta^{(i)}_k$ is the target bearing. The overall measurement vector at time *k* is generated in a similar way. The process model is described in equations (D1), where $a_{x_{k+1}}$ and $a_{y_{k+1}} \sim \mathcal{N}(0, \sigma_a^2)$, Δt is the time discretization step size and N is the total number of targets. The intuition behind the model is to make the targets motion coupled to each other. The target ($i = 1$) is pursued by all other targets ($i > 1$). The changes in the speed and direction of the targets depend on their relative distances to each other. κ_1, κ_2 and κ_3 are the coupling constants in the model. r_t and v_t are the turning radius and velocity respectively and δ is a small offset. The measurement model for the *ith* target is given by equations

(D2). Range measurement noises $v_{r_{k+1}} \sim \mathcal{N}(\mathbf{0}, \mathbf{R_r})$ are mutually correlated but are independent w.r.t. the bearing measurement noises $v_{\theta_{k+1}}$. Bearing measurement noise elements $v^i_{\theta_{k+1}}$ are exponentially distributed with the scale paramter β, such that $\mathbb{E}[(v^i_{\theta_{k+1}})^2] = \beta^2$ and $\mathbb{E}[v^i_{\theta_{k+1}} v^j_{\theta_{k+1}}] = 0$. R_r represent the covariance matrix of $v_{r_{k+1}}$ with $\sigma_r^2 = \mathbb{E}[(v^i_{r_{k+1}})^2]$ and $\sigma^2_{r_x} = \mathbb{E}[v^i_{r_{k+1}} v^j_{r_{k+1}}]$. $\sigma^2_{r_x}$ is assumed to be same for any two targets. The likelihood function can be written as in equation (D3). A more detailed description of the model can be found in [13].

V. Results

We simulate two targets (N =2) in our analysis. Δt is set to 1, σ_a^2 to 0.5 $\mathrm{m}s^{-2}$, σ_r^2 is set to $2000m^2$, $\sigma^2_{r_x}$ to $\frac{3}{10}\sigma_r^2$, while β^2 is set to $\frac{1}{10}\mathrm{rad}^2$. We note that $\sigma_r < D_{i,k}\sigma_\theta \quad \forall i,k$, where $D_{i,k}$ represents the distance of *ith* target from the radar location at time instant k. In this paper, we work only with the strongly coupled model with coupling constants κ_1, κ_2 and κ_3 set to 8000, 0.05 and 0.1 respectively. The turn radius r_t and turn speed v_t are set to 200 m and 10 $\mathrm{m}s^{-1}$ while δ is set to 0.001. DHF and SIR-PF particles are initialized by sampling Gaussian distribution with mean of 20000 m and variance of 5000 m^2 for position elements, while their velocities are sampled from Gaussian distribution with mean and variance of 5 ms^{-1} and 25 m^2s^{-2} respectively. EKF is initialized by sampling the Gaussian with initial state vector as mean and with variances 10^4 and 1 for the position and the velocity respectively. We use root average mean square error (RAMSE) as the performance metric. It is defined as following. Let *M* be the total number of simulation runs for a particular scenario, $x_k^{i,m}$ and $y_k^{i,m}$ denote the positions of the *ith* target along X and Y-axis respectively, at time instant k in the mth trial. Likewise, let $\hat{x}_k^{i,m}$ and $\hat{y}_k^{i,m}$ denote estimated positions for the *ith* target. The RAMSE ϵ_r is then defined as,

$$\epsilon_r = \sqrt{\frac{1}{M}\sum_{m=1}^{M}\left[\frac{1}{2d}\sum_{i=1}^{d}\left(\left(x_k^{i,m} - \hat{x}_k^{i,m}\right)^2 + \left(y_k^{i,m} - \hat{y}_k^{i,m}\right)^2\right)\right]}$$

We have simulated the scenario a total of fifty times ($M = 50$). First, we describe the effect of the NI schemes.

A. Effect of numerical integration schemes

We compare the performance of the four methods mentioned in subsection III-B. While we wrote scripts for the first two methods, MATLAB provided functions *ode23s* and *ode15s* were used for the Rosenbrock and the Gear's methods respectively. We also compare the effect of grid discretization on the performance of the above schemes. We use two specific cases, 10 uniformly spaced pseudo-time points (coarse discretization) and 30 exponentially spaced points (fine discretization). We plot the RAMSE ϵ_r for different schemes in figure 1. We note that the Rosenbrock method with 30 λ points has the lowest RAMSE, while the Euler's scheme with 10 λ points is the worst performer followed closely by the Gears-10. Runge-Kutta methods with both 10 and 30 points are the second best. In fact, the difference in the performance between the two is very small. This is followed by the Gear-30 and the Euler-30 methods. We tabulate the time averaged RAMSE and the average processing time per particle for all methods in the Table I. Note that the time values only represents the time spent while solving the homotopy ODE for a single particle. The largest and the smallest values are highlighted in red and green respectively. It can be seen that while the Rosenbrock-30 is the best method, it is also computationally the most expensive. On the other hand, the Euler-10 is the fastest but the worst performer of all methods. While it is slightly worse than the both RKs and the Gears-30, it is approximately six

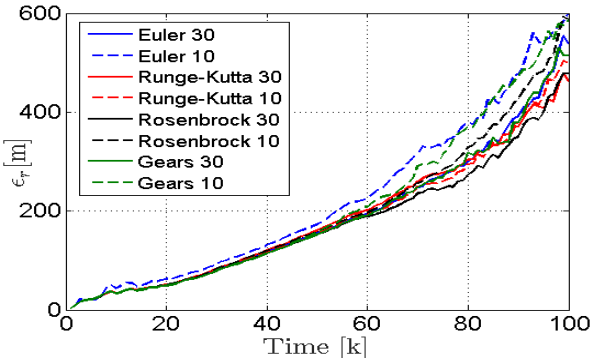

Fig. 1: Comparison of numerical integration schemes

times faster than the RK-30 and more than twelve times as fast as the Rosenbrock-30. Hence, Euler-30 represents a right trade-off between the performance and the processing time.

Method	Avg. Error [m]	Proc.time (pp) [ms]
Euler-30	186.69	5.6
Euler-10	223.07	1.8
Runge-Kutta-30	181.68	38.5
Runge-Kutta-10	184.39	12.7
Rosenbrock-30	173.17	71.9
Rosenbrock-10	196.30	55.8
Gears-30	184.49	26.4
Gears-10	186.69	17.9

TABLE I: Comparison of numerical integration schemes

In the proceeding analysis, we use Euler-30 as the default integration scheme.

B. Effect of shrinkage covariance estimation

Next we analyze the effect of shrinkage estimation schemes. We compare the performance of the six methods mentioned in subsection III-C, together with that of sample and the prior covariance matrices S and P_{EKF} respectively. We describe by the DHF estimate generated using a particular covariance estimation scheme X as DHF-X. We use three metrics to judge the effectiveness of these methods. First and the foremost is the RAMSE of the DHF estimates, which is the central criterion. Second, is the accuracy of the covariance matrix estimates themselves. In the context of the shrinkage estimation, we use the percentage relative improvement in average loss or PRIAL as the measure for the exactness of any shrinkage covariance estimate, following the definition in [17] . This is given by,

$$\text{PRIAL} = \left(1 - \frac{E[||P_{(.)} - P||^2]}{E[||S - P||^2]}\right) \times 100 \qquad (16)$$

where norm $||(.)||$ is the Frobenius norm and S is the sample covariance matrix estimate, while $P_{(.)}$ and P are the shrinked and the true covariance estimates respectively. As P is not known, in the current scenario it is approximated by the covariance estimate from a sampling importance resampling particle filter (SIR-PF) with 25000 particles. Third we use the condition number k_{cond} to analyze the spread in the

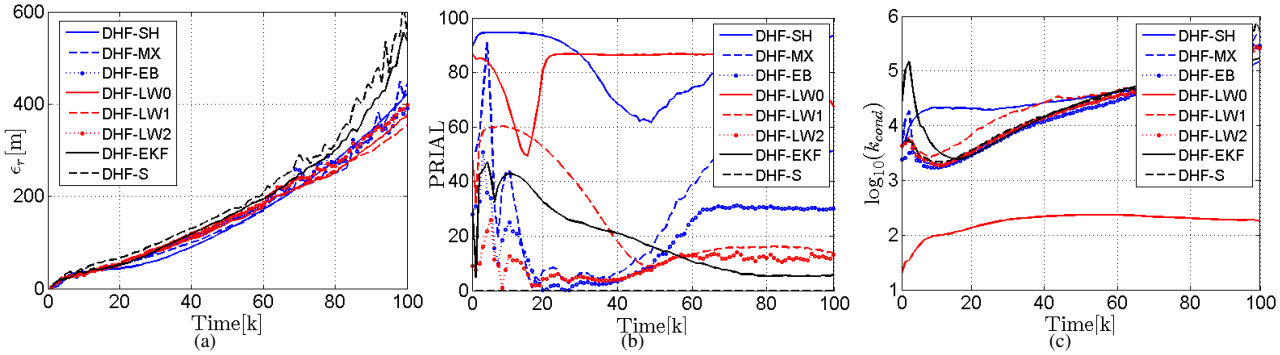

Fig. 2: Comparison of shrinkage estimators (a) RAMSE, (b) PRIAL & (c) Log of condition number

eigenvalues of covariance estimates over the time. Plots for RAMSE, PRIAL and $\log k_{cond}$ are shown in figures 2a, 2b and 2c respectively. Also time averaged values of the measures are tabulated in the Table II. First we discuss the RAMSE for DHF with covariance estimates from all methods. We note that DHF-S is the worst. The S, even though an unbiased estimate, has high variance, which results in relatively high DHF estimation error. DHF-EKF comes next as its error is also wide-off the margin. This can be explained as follow: given that the measurements are non-linear fucntions of state variables, and bearing noise is exponentially distributed, the EKF is not a good approximation for the resulting non-linear and non-Gaussian scenario. Hence the covariance estimates generated by the EKF will not be accurate. DHF-LW0 has the lowest average error amongst all methods. This is because, P_{LW0} is a distribution free estimator, and hence it produces good estimates even in the current non-gaussian scenario. It is followed by the Stein-Haff and Minimax estimators. DHF with the other two covariance estimators from Ledoit and Wolf perform a little inferior relative to the DHF-LW0. Its because, P_{LW1} and P_{LW2} were derived for specific problems in portfolio estimation and have very special structures. This lessens their generality and makes them very application specific. Compared to the DHF-EKF, all estimators except the sample covariance DHF-S have lower average RAMSE. Next we discuss the PRIAL for the covariance estimates. The

Method	Avg. Error [m]	Avg.PRIAL	Ave. Cond. Number
Stein-Haff	159.50	83.30	45080
Minimax	160.38	32.0711	55490
Emp.Bayesian	166.61	18.78	46730
Ledoit-Wolf-0	158.47	81.71	180
Ledoit-Wolf-1	160.91	27.01	53220
Ledoit-Wolf-2	168.54	9.38	48470
EKF covariance	186.69	15.15	67770
Sample covariance	206.28	0	142260

TABLE II: Comparison of covariance estimators

expectation in the formula (16) is calculated by averaging over all simulation runs. A value of 100 means perfect estimation accuracy, while 0 means accuracy as good as the sample covariance matrix S. We note that the PRIAL for P_{SH} is the highest on the average , while is lowest for the P_{LW2}. Again, this can be attributed to the very specific structure of this estimator. PRIAL for P_{LW0}, on the other hand is the most consistent and second best after the P_{SH}. The two also have quite similar time averaged RAMSE. Infact the RAMSE for DHF-SH is lowest for a considerable amount of the total time. As expected, P_{LW0} has the lowest condition number over time, atleast two order of magnitude smaller than all other estimators. S has the highest condition number.

C. *Effect of shrinkage covariance estimation and re-drawing*

Finally, we describe the effect the combined effect of shrinkage and redrawing. The re-drawing was described in Algorithm 2, having parameters $\tilde{a}$, $\tilde{b}$, $\tilde{\nu}$. In our simulations, $\tilde{a}$ was varied between 0 and 1.5, while $\tilde{b}$ between 0.1 and 1. We also tested the case of redrawing in every time step. KDE of the posterior density is carried out using the algorithm mentioned in [24]. The number of GMM components $\tilde{\nu}$ is another parameter of the redrawing algorithm. We varied $\tilde{\nu}$ between 2 and 4. After experimenting with several combinations of $\tilde{a}$, $\tilde{b}$ and $\tilde{\nu}$, we found out that the combination $\tilde{a}$=1.5, $\tilde{b}$ =0.2 and $\tilde{\nu}$=3. has the lowest RAMSE. Also the covariance matrix for the GMM components is estimated using P_{LW0}. Below we compare the performance of EKF, SIR-PF with 1000, 10000 and 25000 particles and different flavors of DHF.

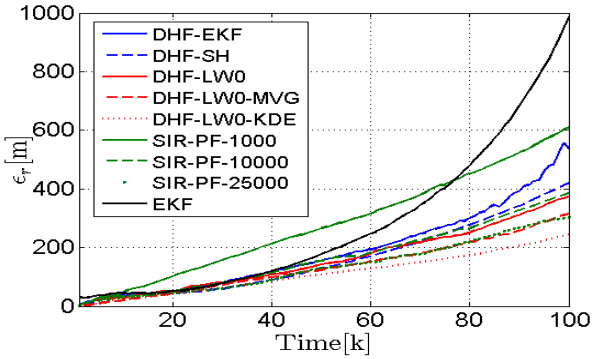

Fig. 3: DHF with redrawing

We first note that the EKF has the highest error and is infact divergent, as the error grows exponentially. Error for the SIR-PF-1000 grows almost linearly with the time, pointing to the inadequacy of the number of particles for the current scenario. Next, we observe that DHF with shrinkage covariances estimation (SH/LW0) is already as good as the particle filter with 10000 particles. Redrawing makes a good situation even better, as it further reduces the estimation error. DHF-LW0-MVG performance is similar to that of SIR-PF with 25000 particles, while DHF-LW0-KDE outperforms all other estimation schemes. Redrawing with KDE has lower error because, instead of approximating the posterior density with a single multivariate Gaussian, KDE uses three components, hence the improved estimation accuracy. Next we compare

the execution time for a single recursion, including both the time and the measurement update steps. Simulations were performed on the computer with Intel Core2 Quad with 2.66 GHz processors and 4 GB RAM. DHF without shrinkage estimation takes 530ms while DHF-LW0 and DHF-SH take additional 0.2 and 0.4 milliseconds respectively. Hence, shrinking covariance estimation puts negligable processing overhead. SIR-PF with 1000, 10000 and 25000 particles take 75ms, 1100ms and 4600 ms respectively. EKF is the fastest of all, taking only 0.4ms per each time step.

VI. Discussion

The Euler method is quite simple, but together with a clever pseudo-time discretization, can perform quite well. It is the most time efficient scheme. We also analyzed different shrinkage estimation schemes. Some of them are tailor made for specific scenarios. The most general one is shrinkage towards identity matrix, where no prior structure of the target matrix is assumed. It is a distribution free approach, and is shown to have outperformed other shrinkage estimators in our analysis. Stein-Haff estimator also gives very good results. Even though it is based on the Gaussian assumption, this estimator works reasonably well in our non-linear and non-Gaussian case. One explaination is that, the isotonic regression which is used to make the covariance PD, still makes the algorithm able handle the non-normality of the data. The estimation of P_{SH} requires the eigendecomposition of the sample covariance S. Doing so could be time consuming for large dimensional systems. Finally, we studied the effect of redrawing on the quality of the filter estimates. First, the density parameters are estimated. Then, the estimated density is used to draw the new set of the particles. The re-drawing algorithm uses several design parameters. We show that the KDE based redrawing combined with the shrinkage estimation gives the best results.

VII. Conclusion

DHF filters, even though not new in the literature, are still not fully explored in detail. They lack the theoretical and numerical analysis that the other contemporary filters have gone through. Especially, the implementational details are ad-hoc and not yet fully formalized. In this paper we have tried to point out the key factors affecting the performance of a standard DHF. Then we have gone through each of them individually and have given the comparison of different approaches. This includes the pseudo-time discretization, different integration schemes, estimation of the prior covariance matrix and the redrawing. Euler's method with exponentially spaced pseudo-time points, provides a nice trade off between the performace and the complexity. DHF with shrinkage covariance estimation is shown to outperform the DHF with the sample covariace matrix and the one with EKF estimate. Finally, it is shown that DHF with properly done redrawing together with the shrinkage estimation can outperform a bootstrap particle filter, with an order of magnitude gain in the processing time.

VIII. Acknowledgment

We acknowledge the support by the EU's Seventh Framework Programme under grant agreement no 607400 (TRAX - Training network on tRAcking in compleX sensor systems) http://www.trax.utwente.nl/.

References

[1] M. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online non-linear/ non-gaussian bayesian tracking," in *IEEE transaction on signal processing*, vol. 50, no. 2. IEEE, February 2002, pp. 174–188.

[2] N. Gordon, D. Salmond, and A. Smith, "Novel approach to nonlinear/non-gaussian bayesian state estimation," in *IEE Proceedings on Radar and Signal Processing.*, Apr 1993, pp. 107–113.

[3] M. Pitt and N. Shephard, "Filtering via simulation: Auxiliary particle filters," in *Journal of the American Statistical Association*, vol. 94, no. 446, 1999.

[4] C. Musso, N. Oudjane, and F. Le Gland, "Improving regularised particle filters," in *Sequential Monte Carlo Methods in Practice*, ser. Statistics for Engineering and Information Science, A. Doucet, N. de Freitas, and N. Gordon, Eds. Springer New York, 2001, pp. 247–271.

[5] F. Daum and J. Huang, "Nonlinear filters with log-homotopy," in *SPIE Proceedings*, September 2007.

[6] F. Daum and J. Huang, "Nonlinear filters with particle flow induced by log-homotopy," in *SPIE Proceedings*, May 2009.

[7] F. Daum and J. Huang, "Particle flow with non-zero diffusion for nonlinear filters," in *SPIE Proceedings*, 2013.

[8] F. Daum and J. Huang, "Numerical experiments on nonlinear filters with exact particle flow induced by log-homotopy," in *Proceeding SPIE*, April 2010.

[9] S. Choi, P. Willet, F. Daum, and J. Huang, "Discussion and application of homotopy filter," in *SPIE Proceedings*, 2011.

[10] T. Ding and M. Coates, "Implementation of Daum-Huang exact flow particle filter," in *IEEE Statistical Signal Processing Workshop (SSP)*, 2012.

[11] K. Bell and L. Stone, "Implementation of the homotopy particle filter in the jpda and map-pf multi-target tracking algorithms," in *Information Fusion (FUSION), 2014 17th International Conference on*, July 2014, pp. 1–8.

[12] P. Bunch and S. Godsill, "The progressive proposal particle filter: Better approximations to the optimal importance density," Tech. Rep., 2014.

[13] M. Khan and M. Ulmke, "Non-linear and non-gaussian state estimation using log-homotopy based particle flow filters," in *Sensor Data Fusion: Trends, Solutions, Applications (SDF), 2014*, Oct 2014, pp. 1–6.

[14] F. Daum and J. Huang, "Nonlinear filters with particle flow," in *SPIE Proceedings*, September 2009.

[15] J. Schäfer and J. Strimmer, "A shrinkage approach to large-scale covariance matrix estimation and implications for functional genomics," 2005.

[16] C. Stein, "Inadmissibility of the usual estimator for the mean of a multivariate normal distribution," in *Proceedings of the Third Berkeley Symposium on Mathematical and Statistical Probability.*, vol. 1, no. 446, 1956, pp. 197—-206.

[17] O. Ledoit and M. Wolf, "Well-conditioned estimator for large-dimensional covariance matrices," *Journal of Multivariate Analysis*, vol. 88, no. 2, pp. 365 – 411, 2004.

[18] O. Ledoit and M. Wolf, "Honey, i shrunk the sample covariance matrix," *The Journal of Portfolio Management*, vol. 30, no. 4, pp. 110–119, 2004.

[19] O. Ledoit and M. Wolf, "Improved estimation of the covariance matrix of stock returns with an application to portfolio selection," *Journal of Empirical Finance*, vol. 10, no. 5, pp. 603 – 621, 2003.

[20] L. Haff, "Emprirical bayes estimation of the multivariate normal covariance matrix," *Ann. Statist.*, vol. 8, no. 3, pp. 586–597, 1980.

[21] C. Stein, "Estimation of a covariance matrix, rietz lecture," 39th Annual Meeting IMS, Atlanta, GA, Tech. Rep., 1975.

[22] S. Lin and M. Perlman, "A monte carlo comparison of four estimators of a covariance matrix," Tech. Rep. 44, 1984.

[23] C. Stein, "Series of lectures given at the university of washington," Seattle, Tech. Rep., 1982.

[24] M. Kristan, A. Leonardis, and D. Skočaj, "Multivariate online kernel density estimation with gaussian kernels," *Pattern Recognition*, vol. 44, pp. 2630–2642, 2011.