METROPOLIS-HASTINGS IMPROVED PARTICLE SMOOTHER AND MARGINALIZED MODELS

Jerker Nordh

Lund University
Department of Automatic Control
SE-221 00 Lund, Sweden (e-mail: jerker.nordh@control.lth.se)

ABSTRACT

This paper combines the Metropolis-Hastings Improved Particle Smoother (MHIPS) with marginalized models. It demonstrates the effectiveness of the combination by looking at two examples; a degenerate model of a double integrator and a fifth order mixed linear/nonlinear Gaussian (MLNLG) model. For the MLNLG model two different methods are compared with the non-marginalized case; the first marginalizes the linear states only during the filtering, the second marginalizes during both the foward filtering and backward smoothing pass. The results demonstrate that marginalization not only improves the overall performance, but also increases the rate of improvement for each iteration of the MHIPS algorithm. It thus reduces the required number of iterations to beat the performance of a Forward-Filter Backward Simulator approach for the same model.

Index Terms— Metropolis-Hasting Improved Particle Smoother, Rao-Blackwellized smoothing, Particle Smoothing, Particle Filter

1. INTRODUCTION

During the last decade particle filters have become popular for solving nonlinear estimation problems. For linear Gaussian systems the Kalman filter [1] provides the optimal estimate, and some extensions such as the Extended Kalman Filter [1] and Unscented Kalman Filter [2] have been proposed to handle nonlinear systems. These make simplifying assumptions such as that the system can be locally approximated by a linearized model or that the resulting distribution will be Gaussian. For models where these assumptions do not hold particle filters [3] can provide superior performance, some examples are multi-target tracking [4] and Simultanous Localization and Mapping (SLAM) [5]. Instead of assuming a Gaussian distribution the true distribution is approximated using a set of weighted point estimates, or particles,

$$\hat{p}(x_t|y_{1:t}) = \sum_{i=1}^{N} w_t^{(i)} \delta(x_t - x_t^{(i)}) \tag{1}$$

where $x_t^{(i)}$ are particles and $w_t^{(i)}$ the weights. As the number of particles increases the approximation approaches the true distribution.. A typical particle filter implementation is shown in algorithm 1, a common choice of proposal distribution is $r(x_{t+1}|x_t, y_{t+1}) = p(x_{t+1}|x_t)$ which leads to some simplifications during the computations. For a more detailed introduction to the particle filter see [3]. A drawback with particle filters is that the number of particles needed increases with the dimension of the problem [6] [7], because of this it is of interest to marginalize over some of the states if possible to reduce the dimension of the estimation problem and thus the computational complexity. A typical example of this are models where some of the states only occur linearly and are only affected by additive Gaussian noise. Conditioned on the remainder of the states those could thus be optimally estimated using a Kalman filter. Filters exploiting this is typically referred to as Rao-Blackellized Particle Filters (RBPF) [8].

Algorithm 1 Particle Filter

for $i = 1$ to N **do**
  Sample $x_1^{(i)} \sim r(x_1)$.
  Set $\tilde{w}_1^{(i)} = p(x_1^{(i)})p(y_1|x_1^{(i)})/r(x_1^{(i)})$
end for
Set $w_1^{(i)} = \tilde{w}_1^{(i)}/(\sum_{j=1}^{N} \tilde{w}_1^{(j)}), \quad \forall i \in [1, N]$.
for $t = 2$ to T **do**
  for $i = 1$ to N **do**
    Sample ancestor indices, a_t^i, with $\mathbb{P}\left(a_t^i = j\right) = w_{t-1}^{(j)}$.
    Sample $x_{t+1}^{(i)}$ from $r(x_{t+1}|x_t^{(a_t^i)}, y_{t+1})$
    Calculate weights:
    $\tilde{w}_{t+1}^{(i)} = p(y_{t+1}|x_{t+1}^{(i)})p(x_{t+1}|x_t^{(a_t^i)})/r(x_{t+1}|x_t^{(a_t^i)}, y_{t+1})$
  end for
  Set $w_t^{(i)} = \tilde{w}_t^{(i)}/(\sum_{j=1}^{N} \tilde{w}_t^{(j)}), \quad \forall i \in [1, N]$.
end for

Conceptually by storing the ancestral paths of the particles the filter also provides a smoothed state estimate, i.e

The author is a member of the LCCC Linnaeus Center and the eLLIIT Excellence Center at Lund University. The author would like to thank professor Bo Bernhardsson, Lund University, for the feedback provided on this work.

978-0-9928626-3-3/15/$31.00 ©2015 IEEE

$p(x_t|y_{1:T})$ where $t < T$. However, due to the resampling step in the particle filter, in practice for all $t \ll T$ the ancestral paths will all be identical, thus providing a very poor approximation of the true posterior distribution [9]. Because of this a number of particle smoothing algorithms have been proposed, the most commonly used is the Forward Filter Backward Simulator (FFBSi). The FFBSi complements the particle filter by performing a backwards sweep where the ancestor of each particle is sampled using updated weights that depend on both the old filter weights, as well as the probability $p(x_{t+1}|x_t)$. For a more thorough introduction to particle smoothing see [10] and for the Rao-Blackwellized particle smoothing see [11].

A weakness with the FFBSi is that it only reuses the samples from the forward filter, this is something the Metropolis-Hastings Backward Proposer (MHBP) [12] and Metropolis-Hastings Improved Particle Smoother (MHIPS) addresses. The rest of this paper will focus on how to combine MHIPS with marginalized models, showing how to combine it with a previously published method [13] [11] for marginalization of mixed linear/nonlinear Gaussian models (MLNLG). First it briefly discusses the, in general non-Markovian, filtering/smoothing problem then shows a trivial example of a model where the computational cost of the general solution can be avoided through the introduction of extra variables. These variables are propagated backwards during the smoothing step in similar manner as the mean and covariance are propagated forward in a RBPF. Section 2 introduces the example models used in this article, section 3 discussed MHIPS in detail and shows how to extend it for marginalized models and section 4 presents some results of applying the method to the example models and finally section 5 summarizes the paper.

2. MODELS

Two models are used in this paper, a degenerate double integrator and a fifth order MLNLG model. The double integrator is included to illustrate how the methods are affected by degeneracy and to introduce the concept used for the marginalization of MLNLG models in a simpler setting. The main issue for working with non-Markovian models is the need to evaluate densities of the form $p(x_{t+1:T}, y_{t+1:T}|x_{1:t}, y_{1:t})$, typically the complexity of this operation grows with the length, T, of the dataset. The following subsections shows how this can be accomplished in constant time for the models considered in this paper.

2.1. Double integrator

Due to the noise only acting on the input there is a deterministic relation between the two states making the model degenerate and not suitable for the standard particle smoothing methods. This coupling means that $p(x_{t+1}^{(i)}|x_t^{(j)}) = 0, \quad \forall j \neq a_i$ where a_i is the index of the ancestor for particle $x_{t+1}^{(i)}$.

$$\begin{aligned} x_{t+1} &= \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x_t + \begin{pmatrix} 0 \\ 1 \end{pmatrix} w_t && \text{(2a)} \\ y_t &= \begin{pmatrix} 1 & 0 \end{pmatrix} x_t + e_t && \text{(2b)} \\ w_t &\sim N(0, Q), \quad e_t \sim N(0, R) && \text{(2c)} \end{aligned}$$

The model in (2) can be rewritten as a first order system with a non-Markovian structure, for notational brevity the notation $x_t = (p_t \quad v_t)^T$ is introduced and the model can then be rewritten as

$$\begin{aligned} v_{t+1} &= v_t + w_t && \text{(3a)} \\ y_t &= p_0 + \sum_{i=0}^{t-1} v_i + e_t && \text{(3b)} \\ w_t &\sim N(0, Q), \quad e_t \sim N(0, R) && \text{(3c)} \end{aligned}$$

The estimation problem could now be solved using a non-Markovian particle smoother [14]. At a quick glance this looks like simply reintroducing the p-state from the original model, but the key distinction is that this new variable is a function of the past trajectory, and not included as a state in the model. During the filtering each particle also stores the corresponding sum associated with its past trajectory.

$$\begin{aligned} v_{t+1} &= v_t + w_t && \text{(4a)} \\ s_{t+1} &= s_t + v_t && \text{(4b)} \\ y_t &= s_t + e_t && \text{(4c)} \\ s_0 &= p_0 && \text{(4d)} \\ w_t &\sim N(0, Q), \quad e_t \sim N(0, R) && \text{(4e)} \end{aligned}$$

The (non-Markovian) smoother will need to evaluate the probability density (5a), but only up to proportionality which allows it to be rewritten as (5b).

$$\begin{aligned} & p(y_{t+1:T}, v_{t+1:T}|v_{1:t}, y_{1:t}) && \text{(5a)} \\ & = p(y_{t+1:T}|v_{1:T}, y_{1:t}) p(v_{t+1:T}|v_{1:t}) \\ & \underset{v_{1:t}}{\propto} p(y_{t+1:T}|v_{1:T}) p(v_{t+1}|v_t) && \text{(5b)} \end{aligned}$$

Utilizing the same approach as the authors of [13] and noticing that (5b) only needs to be evaluated up to proportionality (with regard to $v_{1:t}$) it is possible to propagate information backwards during the smoothing in the same way as the s_t variables propagates the sum during filtering. The first factor of (5b) can be evaluated up to proportionality as follows

$$\begin{aligned} & p(y_{t+1:T}|s_t, v_{t:T}) = \prod_{k=t+1}^{T} p(y_k|s_t, v_{t:k-1}) && \text{(6a)} \\ & \underset{s_t, v_t}{\propto} \prod_{k=t+1}^{T} e^{(s_t+v_t)^2 - 2(y_k - \sum_{j=t+1}^{k-1} v_j)(s_t+v_t)} && \text{(6b)} \\ & = e^{(T-t)(s_t+v_t)^2 - 2\sum_{k=t+1}^{T}(y_k - \sum_{j=t+1}^{k-1} v_j)(s_t+v_t)} && \text{(6c)} \end{aligned}$$

Through the introduction of two new variables N_t, γ_t that are propagated backwards during the smoothing this allows (6) to be evaluated as

$$\log p(y_{t+1:T}|s_t, v_t, v_{t+1:T}) + \text{constant} = \frac{1}{2R}(N_{t+1}(s_t+v_t)^2 - 2\gamma_{t+1}(s_t+v_t)) \quad (7a)$$

$$N_t = N_{t+1} + 1, \quad N_T = 1 \quad (7b)$$

$$\gamma_t = \gamma_{t+1} + y_t - N_{t+1}v_t, \quad \gamma_T = y_T \quad (7c)$$

Using (7) it is now possible to evaluate the required smoothing density in constant time.

2.2. Mixed Linear/Nonlinear Gaussian

This model was introduced in [11] as an extension to the commonly used standard nonlinear model, the difference is that the constant 25 has been replaced by the output of a fourth order linear system.

$$\xi_{t+1} = 0.5\xi_t + \theta_t\frac{\xi_t}{1+\xi_t^2} + 8\cos 1.2t + v_{\xi,t} \quad (8a)$$

$$z_{t+1} = \begin{pmatrix} 3 & -1.691 & 0.849 & -0.3201 \\ 2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0.5 & 0 \end{pmatrix} z_t + + v_{z,t} \quad (8b)$$

$$y_t = 0.05\xi_t^2 + e_t \quad (8c)$$

$$\theta_t = 25 + \begin{pmatrix} 0 & 0.04 & 0.044 & 0.008 \end{pmatrix} z_t \quad (8d)$$

$$\xi_0 = 0, \quad z_0 = \begin{pmatrix} 0 & 0 & 0 & 0 \end{pmatrix}^T \quad (8e)$$

$$v_{\xi,t} \sim N(0, Q_\xi),\ v_{z,t} \sim N(0, Q_z) \quad (8f)$$

$$e_t \sim N(0, R) \quad (8g)$$

The z-states are not fully observable, but the affine combination θ is. The four z-states appear affinely and given the trajectory $\xi_{1:T}$ the remaining estimation problem could easily be solved using a regular Kalman filter. Several solutions have been proposed to utilize this fact when performing particle smoothing, in this paper two of these are compared. In the forward step both methods work the same way, marginalizing the linear states by computing the sufficient statistics, i.e. the mean value ($\bar{z}_t$) and the covariance (P_t), instead of sampling the z-states. This allows the densities $p(\xi_{t+1}|\xi_{1:t}, y_{1:t})$ and $p(y_t|\xi_{1:t}, y_{1:t-1})$ to be expressed as $p(\xi_{t+1}|\xi_t, \bar{z}_t, P_t)$ and $p(y_t|\xi_t, \bar{z}_t, P_t)$ respectively. The differences are in the backward smoothing recursions; the first method [11] samples the linear states during the smoothing step, the second instead fully marginalizes the model resulting in a non-Markovian smoothing problem [13]. The first method thus effectively only uses marginalization during the filtering. The smoothing for the second method is accomplished using the idea of propagating information backwards that was demonstrated in section 2.1, for details the reader is referred to the original paper [13].

3. ALGORITHM

This section shows how to combine MHIPS [15] with the type of marginalized models shown in section 2. Algorithm 2 summarizes the MHIPS algorithm for Markovian models. It is initialized with the ancestral trajectories, denoted $\tilde{x}_{1:T}$, from the forward particle filter, it then iterates over the trajectories from end to beginning R times, and for every time-step a new sample, x', is proposed from the proposal density $q(x'_t|\tilde{x}_{t+1}, y_t, \tilde{x}_{t-1})$. This new sample is accepted with probability given by

$$1 \wedge \frac{p(\tilde{x}_{t+1}|x'_t)}{p(\tilde{x}_{t+1}|\tilde{x}_t)}\frac{p(y_t|x'_t)}{p(y_t|\tilde{x}_t)}\frac{p(x'_t|\tilde{x}_{t-1})}{p(\tilde{x}_t|\tilde{x}_{t-1})}\frac{q(\tilde{x}_t|\tilde{x}_{t+1}, y_t, \tilde{x}_{t-1})}{q(x'_t|\tilde{x}_{t+1}, y_t, \tilde{x}_{t-1})}. \quad (9)$$

where $(a \wedge b)$ denotes the minimum value of a and b. When a sample is accepted it replaces the previous value in $\tilde{x}_{1:T}$ by setting $\tilde{x}_t = x'_t$. For the non-Markovian case the proposal and acceptance probabilities have to be extended in the same way as for the backward simulator type of smoother [16] [9]. This results in the proposal density $q(x'_t|\tilde{x}_{t+1:T}, y_{1:T}, \tilde{x}_{1:t-1})$ and the acceptance probability is now given by

$$1 \wedge \frac{p(\tilde{x}_{t+1:T}, y_{t+1:T}|x'_t, \tilde{x}_{1:t-1}, y_{1:t})}{p(\tilde{x}_{t+1:T}, y_{t+1:T}|\tilde{x}_t, \tilde{x}_{1:t-1}, y_{1:t})} \times \frac{p(y_t|x'_t, \tilde{x}_{1:t-1}, y_{1:t-1})}{p(y_t|\tilde{x}_t, \tilde{x}_{1:t-1}, y_{1:t-1})}\frac{p(x'_t|\tilde{x}_{1:t-1}, y_{1:t-1})}{p(\tilde{x}_t|\tilde{x}_{1:t-1}, y_{1:t-1})} \times \frac{q(\tilde{x}_t|\tilde{x}_{t+1:T}, y_{1:T}, \tilde{x}_{1:t-1})}{q(x'_t|\tilde{x}_{t+1:T}, y_{1:T}, \tilde{x}_{1:t-1})}. \quad (10)$$

At each time-step t, the MHIPS is choosing between two complete trajectories $x_{1:t}$, they are however identical except for the last state (x_t). Algorithm 3 gives the extended algorithm exemplified using the propagation of backwards variables as derived for the marginalized double integrator in (7), the concept is the same for other models where the necessary information can be back-propagated in a similar manner. For models where that is not possible it is of course possible to evaluate the required densities by iterating over the future parts of the trajectory, however the poor scaling properties of that approach makes it an unattractive method.

Algorithm 2 MHIPS

1: Sample $\{\tilde{x}_{1:T}^{(j)}\}_{j=1}^M$ from the ancestral paths of the forward filter, drawing each trajectory with probability $\omega_T^{(i)}$.
2: **for** $r = 1$ to R **do**
3: **for** $t = T$ to 1 **do**
4: **for** $j = 1$ to M **do**
5: Sample $x'^{(j)}_t \sim q_t(x_t|\tilde{x}^{(j)}_{t+1}, y_t, \tilde{x}^{(j)}_{t-1})$
6: With probability given by (9) set $\tilde{x}^{(j)}_t = x'^{(j)}_t$
7: **end for**
8: **end for**
9: **end for**

Algorithm 3 MHIPS non-Markov

Sample $\{\tilde{x}_{1:T}^{(j)}\}_{j=1}^{M}$ from the ancestral paths of the forward filter, drawing each trajectory with probability $\omega_T^{(i)}$.
for $r = 1$ to R **do**
  for $t = T$ to 1 **do**
    for $j = 1$ to M **do**
      Sample $x'^{(j)}_t \sim q_t(x_t|\tilde{x}^{(j)}_{t+1:T}, y_t, \tilde{x}^{(j)}_{1:t-1})$
      With probability given by (10) set $\tilde{x}^{(j)}_t = x'^{(j)}_t$
      Calculate backward propagating variables $(N_t^{(j)}, \gamma_t^{(j)})$
    end for
  end for
  for $t = 1$ to T **do**
    for $j = 1$ to M **do**
      Update forward propagating variables $(s_t^{(j)})$
    end for
  end for
end for

4. RESULTS

All the experiments have been done using the proposal density $q(x'_t|\tilde{x}_{t+1:T}, y_{1:T}, \tilde{x}_{1:t-1}) = p(x'_t|y_{1:t-1}, \tilde{x}_{1:t-1})$ for the marginalized case and $q(x'_t|\tilde{x}_{t+1}, y_t, \tilde{x}_{t-1}) = p(x'_t|\tilde{x}_{t-1})$ for the Markovian case. The Python source code for the examples can be downloaded from [17], all simulation have been done using the pyParticleEst [18] software framework.

4.1. Double integrator

The methods are tested against an example with $Q = R = 1$ and initial state $x_1 = (-10 \quad 1)^{\mathrm{T}}$ and using 50 particles in the forward filter and 10 smoothed trajectories. Fig. 1 shows the RMSE for the double integrator example as a function of the number of iterations of the MHIPS algorithm. As expected there is no improvement for the degenerate case. For the marginalized case there is clear improvement using MHIPS and after only 9 iterations it yields better average RMSE than the FFBSi smoother. It would of course have been possible to initialize the MHIPS algorithm using the trajectories obtained from the FFBSi smoother instead of the ancestral paths from the particle filter, giving a lower initial RMSE, but to clearly demonstrate the initial rate of improvement of the MHIPS algorithm this was not done.

4.2. MLNLG model

The methods are tested against an example with $Q_e = 0.005$, $Q_z = 0.01 I_{4\text{x}4}$, $R = 0.1$ and initial state $(\xi_1^{\mathrm{T}} \quad z_1^{\mathrm{T}})^{\mathrm{T}} = 0_{5\text{x}1}$ and using 100 particles in the forward filter and 10 smoothed trajectories. Fig. 2 shows the average RMSE of the ξ-state as a function of the number of iterations of the MHIPS algorithm, Fig. 3 shows the average RMSE of θ, the affine function of the z-states. Both figures include results for three different methods; the case when all the states are sampled, when the z-states are sampled during the smoothing step and finally when the z-states are fully marginalized for both the filter and smoother. The figures also include the performance of a FFBSi smoother for all three cases as a reference. It can be seen that marginalization as expected leads to a lower average RMSE for both FFBSi and MHIPS. It also shows a higher rate of improvement when using MHIPS and marginalization, requiring fewer iterations to beat the performance of FFBSi. Suggesting that proper marginalization is even more important for MHIPS than for FFBSi.

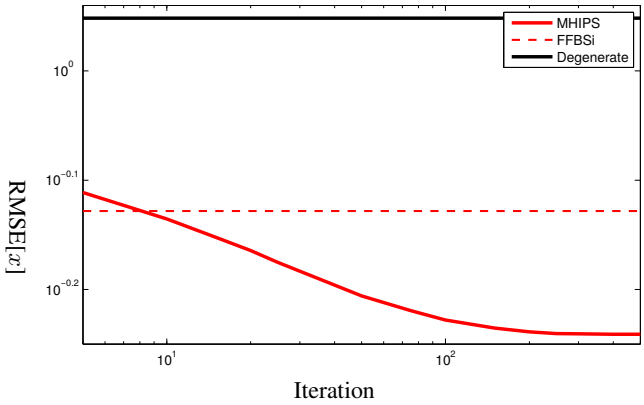

Fig. 1. Average RMSE over 10000 realizations in logarithmic scale. The dotted black line is for the degenerate model, as expected it shows no improvement using MHIPS. The solid red line is for MHIPS on the marginalized model and dashed line when using FFBSi

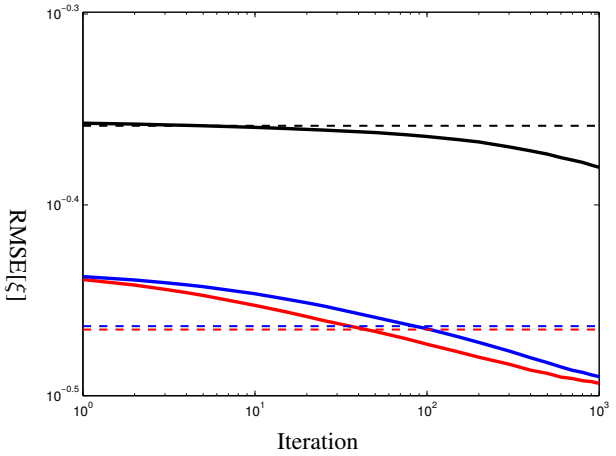

Fig. 2. Average RMSE over 1500 realisation for ξ in logarithmic scale. The black (top) solid line is for the non-marginalized model, the blue (middle) solid line is for the model marginalized in the forward direction and the red (bottom) solid line is for the fully marginalized model. The dashed lines with corresponding colors (and order) are the RMSE obtained when using FFBSi.

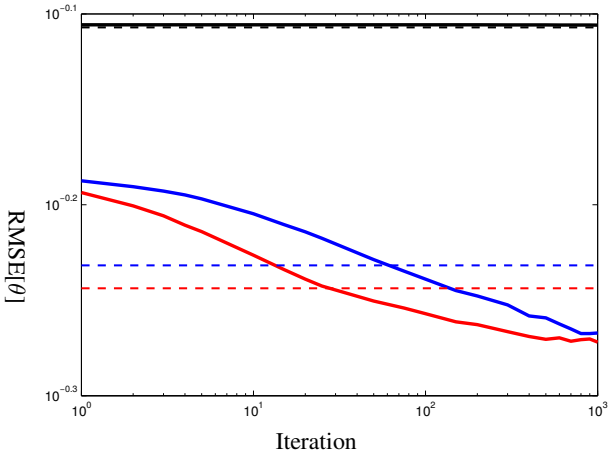

Fig. 3. Average RMSE over 1500 realisation for θ in logarithmic scale. The black (top) solid line is for the non-marginalized model, the blue (middle) solid line is for the model marginalized in the forward direction and the red (bottom) solid line is for the fully marginalized model. The dashed lines with corresponding colors (and order) are the RMSE obtained when using FFBSi.

5. CONCLUSION

This article has demonstrated how to combine MHIPS with marginalized models and combines it with the method for fully marginalizing the linear sub-states in a MLNLG model using the approach presented in [13]. It compares the marginalized MHIPS with the regular MHIPS, showing the importance of marginalization with MHIPS since it enables MHIPS to outperform the FFBSI smoother using a lower number of iterations.

REFERENCES

[1] Greg Welch and Gary Bishop, "An introduction to the kalman filter," 1995.

[2] S.J. Julier and J.K. Uhlmann, "Unscented filtering and nonlinear estimation," *Proceedings of the IEEE*, vol. 92, no. 3, pp. 401–422, 2004.

[3] Arnaud Doucet, Simon Godsill, and Christophe Andrieu, "On sequential monte carlo sampling methods for bayesian filtering," *Statistics and Computing*, vol. 10, no. 3, pp. 197–208, 2000.

[4] Kenji Okuma, Ali Taleghani, Nando De Freitas, James J Little, and David G Lowe, "A boosted particle filter: Multitarget detection and tracking," in *Computer Vision-ECCV 2004*, pp. 28–39. Springer-Verlag, 2004.

[5] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al., "Fastslam: A factored solution to the simultaneous localization and mapping problem," in *AAAI/IAAI*, 2002, pp. 593–598.

[6] Alexandros Beskos, Dan Crisan, Ajay Jasra, et al., "On the stability of sequential monte carlo methods in high dimensions," *The Annals of Applied Probability*, vol. 24, no. 4, pp. 1396–1445, 2014.

[7] Patrick Rebeschini and Ramon Van Handel, "Can local particle filters beat the curse of dimensionality?," *arXiv preprint arXiv:1301.6585*, 2013.

[8] T. Schön, F. Gustafsson, and P.-J. Nordlund, "Marginalized particle filters for mixed linear/nonlinear state-space models," *Signal Processing, IEEE Transactions on*, vol. 53, no. 7, pp. 2279–2289, 2005.

[9] Fredrik Lindsten and Thomas B Schön, "Backward simulation methods for monte carlo statistical inference," *Foundations and Trends in Machine Learning*, vol. 6, no. 1, pp. 1–143, 2013.

[10] Mark Briers, Arnaud Doucet, and Simon Maskell, "Smoothing algorithms for state-space models," *Annals of the Institute of Statistical Mathematics*, vol. 62, no. 1, pp. 61–89, 2010.

[11] F. Lindsten and T.B. Schön, "Rao-Blackwellized Particle Smoothers for mixed linear/nonlinear state-space models," Tech. Rep., 2011.

[12] Pete Bunch and Simon Godsill, "Improved particle approximations to the joint smoothing distribution using markov chain monte carlo," *Signal Processing, IEEE Transactions on*, vol. 61, no. 4, pp. 956–963, 2013.

[13] Fredrik Lindsten, Pete Bunch, Simon J Godsill, and Thomas B Schon, "Rao-Blackwellized Particle Smoothers for mixed linear/nonlinear state-space models," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*. IEEE, 2013, pp. 6288–6292.

[14] Fredrik Lindsten and Thomas B. Schn, "Backward simulation methods for monte carlo statistical inference," *Foundations and Trends in Machine Learning*, vol. 6, no. 1, pp. 1–143, 2013.

[15] Cyrille Dubarry and Randal Douc, "Particle approximation improvement of the joint smoothing distribution with on-the-fly variance estimation," *arXiv preprint arXiv:1107.5524*, 2011.

[16] Fredrik Lindsten, Michael I. Jordan, and Thomas B. Schön, "Particle Gibbs with ancestor sampling," *Journal of Machine Learning Research*, vol. 15, pp. 2145–2184, 2014.

[17] Jerker Nordh, "source code: http://www.control.lth.se /staff/jerkernordh/mhips-marginalized.html," 2015.

[18] Jerker Nordh, "pyParticleEst: A Python framework for particle based estimation methods," *Journal of Statistical Software*, 2015, Provisionally accepted.