Bayesian Inference for Thermal Model of Synchronous Generator Part–II: State Estimation

MADHUSUDHAN PANDEY AND BERNT LIE (Member, IEEE)
Telemark Modeling and Control Center (TMCC), 3901 Porsgrunn, Norway
Corresponding author: Bernt Lie (bernt.lie@usn.no)

ABSTRACT
This paper is the Part II of the series on parameter and state estimation using the Bayesian inference for a thermal model of a synchronous generator. Part I is about Parameter Estimation. In this paper, state estimation of rotor copper and air-gap temperatures of the synchronous generator are performed using Bayesian inference. Estimators such as Unscented Kalman Filter (UKF), Ensemble Kalman Filter (EnKF), and Particle Filters (PFs) with different sampling algorithms, are compared based on the estimation accuracy and computational time. The inferences are drawn for the posterior distributions of the state, dispersion of particles, error convergence and particle realizations of the state estimator, choice, and the computational effort of the estimators. Results show that UKF has fair estimation accuracy with the fastest computational time as compared to other estimators.

INDEX TERMS
Bayesian method, EnKF, particle filters, state estimation, synchronous generator, thermal model, UKF.

I. INTRODUCTION
Because of the growing usage of intermittent power sources, such as solar and wind power, in a shared electrical grid, dispatchable sources such as hydro power should be able to eliminate the unpredictability in load and generation induced by the intermittent sources. As a result, for best active power utilization, hydro generators should be able to run beyond their thermal capability limit. This necessitates the monitoring of the hydro generator’s interior temperatures. In Part I of this paper titled “Bayesian Inference for Thermal Model of Synchronous Generator Part I : Parameter Estimation”, various aspects of parameter estimation and parameter identifiability for the thermal model of the synchronous generator is emphasized. In this paper, for the same thermal model of synchronous generator, various aspects of state estimation using Bayesian inference is emphasized. It is requested to follow Part I for the governing equations, and experimental data of the thermal model of the air-cooled synchronous generator.

Section II outlines materials and methods. Section III provides state estimation using Bayesian inference. In Section IV, particle filters are initialized and implemented. Furthermore, PFs and Kalman filters are compared in Section V. Section VI provides estimation versus computational time of the estimators. Conclusions are drawn in Section VII.

II. MATERIALS AND METHODS
A. SYSTEM DESCRIPTION
Figure 1 shows the Bayesian framework for the inferences about the states of a dynamical system. In the figure, \( x, u, z, \theta \) and \( y \) are the states, inputs, algebraic variables, parameters and outputs, respectively. For the state estimation shown in Fig. 1, \( x \) and \( z \) are estimated when \( u, \theta \) and \( y \) are given. \( p(\mathbf{x}_1) \) is the prior distribution of the initial conditions of the states, \( p(\mathbf{Y}_1 | \mathbf{x}_1) \) is the initial likelihood and \( p(\mathbf{x}_k | \mathbf{Y}_k) \) is the estimated posterior distribution of states by formulating a recursive relation considering measurement available at each instance of measurement data for outputs where \( \mathbf{Y}_k = [y_1, y_2, \ldots, y_k] \). In the figure, we see that the Bayesian state estimation allows posterior distribution of the state, dispersion analysis of the state, particles realization needed in the state estimation, relationship between the parameters with the multimodal posterior distribution and the choice of particle or ensemble type filters, inference on particles realization with error convergence and computational effort, and sub-optimality comparison between different estimators.
For a state estimation for measurement \( y \) available function

In practice, the analytical solution for the posterior density

a recursive state estimation problem can be formulated.

The governing equations contain 3 differential equations representing time evolution for the metal temperatures, and 3 algebraic equations relating the metal temperatures with the air temperatures.

III. STATE ESTIMATION

A. RECURSIVE BAYESIAN FILTER

The Bayesian formulation of the parameter estimation problem as in Part I is also used for state estimation by replacing parameter \( \theta \) with state \( x \). However, states are quantities that change with time unlike parameters, the state estimation is only done at the point where the time is specified, let’s say at time instant \( k \). This estimate is usually a point estimate though the posterior density distribution of state is also available at each instance of time.

Furthermore, the measurement \( y \) as in the parameter estimation problem (Part I) is replaced with the measurement available \( Y_k = [y_1, y_2, \ldots, y_k] \). The Bayesian formulation at time instant \( k \) conditioned with measurement up to \( Y_k \) is done as

\[
p(x_k | Y_k) = \frac{p(y_k | x_k) p(x_k | Y_{k-1})}{p(y_k | Y_{k-1})}
\]

(1)

where \( p(y_k | x_k) \) and \( p(y_k | Y_{k-1}) \) are the likelihood and the evidence known from the information of the measurement noise density, \( p(x_k | Y_{k-1}) \) is the prior and calculated as

\[
p(x_k | Y_{k-1}) = \int p(x_k | x_{k-1}) p(x_{k-1} | Y_{k-1}) dx_{k-1}.
\]

(2)

For a state estimation for measurement \( y = [y_1, y_2, \ldots, y_N] \) with the initial state density function given as

\[
p(x_1 | Y_1) = \frac{p(Y_1 | x_1) p(x_1 | Y_0)}{p(Y_1)},
\]

(3)

a recursive state estimation problem can be formulated. In practice, the analytical solution for the posterior density function \( p(x_k | Y_k) \) given by Eq. (1) exists for only few special cases. The analytical derivation considering a linear system with additive Gaussian noise leads to the optimal linear Kalman filter. Further information about the Kalman filter and its derivatives for state estimation can be found in [3]. However, for a non-linear system and systems where analytical solutions are impossible, we formulate a numerical solution considering a sample drawn from the known initial states’ density function and recursively obtain the numerical solution using Eqs. (1-3) for the posterior distribution.

B. PARTICLE FILTER

The numerical implementation of the recursive Bayesian filter acquired drawing of random values from a known initial density function for the state \( p(x_1) \). Each random value is considered a particle. Using these particles for the numerical approximation of the posterior of the Bayesian estimator is often termed particle filters. It is interesting to note that the estimation accuracy increases as the number of particles increases. Hence, there is always a trade-off between computational speed and estimation accuracy.

The particle filter algorithm taken from [3] is given in Table 1. \( p(x_1), p(w_1), \) and \( p(v_1) \) are the initial prior densities of the states, process noise, and the measurement noise. For \( i = 1 \): \( N_p \) \( x_{i|k-1} \) are \( a \) priori particles obtained by propagating the prior particles through the process equations. Similarly, \( y_{i|k-1} \) are the predicted measurements use for calculating the innovation \( e_i \). The innovation is used for calculating the likelihood \( q_i \). A simple expression for the likelihood is found considering measurement noise as \( v_k \sim N(0, \mathcal{V}) \) i.e. with zero-mean \( \overline{v} = 0 \) and co-variance \( \mathcal{V} \). After then the likelihood is normalized. The normalized likelihood are then resampled to obtain the posterior density. There are several particle methods for resampling. The particles drawn from the posterior distribution are called as \( a \) posteriori estimates and considered as the sub-optimal estimates. The optimality of the estimates increases as the number of particles are increased.

The \( a \) priori particles are distributed with uneven weights. The unevenly weighted particles produce degeneracy which will eventually cause estimation with larger variance. Particle degeneracy is actually a situation where a few particles having lower/higher weights dominate other particles. This gives rise to resampling and is done to make sure all particles have equal weights to minimize the estimation variance. To eradicate the problem of degeneracy, particles with lower weights are replaced with higher weights to evenly distribute particles on the basis of their weights. This process of distribution of particles evenly on the basis of weights is often termed as resampling. The resampling is done based on the resampling quality, computational cost, and easy implementation. The resampled particles are called as \( a \) posteriori particles from which any statistical moments, like mean and standard deviation, is updated to recursively run the filter.

C. RESAMPLING

A simple resampling algorithm is taken from [4], given in Table 1, and this algorithm is named as Ris04 for distinguishing it from other algorithms. In Ris04 algorithm we first generate a uniformly distributed random number \( r \) as \( r \sim \mathbb{U}(0,1] \). Then we find the cumulative sum of the likelihood \( q^+ \) in each iteration. If \( q^+ \) is greater than equal to the generated random number we then set the \( a \) priori particles as...
TABLE 1. Generalized particle filter.

| Initialization, $k = 1$: |
|--------------------------|
| Draw $N_p$ particles for states, process and measurement noise from initial probability distribution $p(x_1)$, $p(w_1)$ and $p(v_1)$. |
| $x_{1|1}^{k} \sim p(x_1)$, $i \in \{1, 2, ..., N_p\}$ |
| $w_{1}^{k} \sim p(w_1)$, $i \in \{1, 2, ..., N_p\}$ |
| $v_{1}^{k} \sim p(v_1)$, $i \in \{1, 2, ..., N_p\}$ |
| for $k = 2, 3, ...$ |

Propagation step:

(a priori particles)

1. Propagate a priori particles through measurement model

2. Calculate relative likelihood $q_i$, i.e. $p(y_k | x_{k|k-1})$ from innovation

3. Normalize $q_i$

4. Resampling for a posteriori estimate

5. A posteriori update

| Information update: |
|---------------------|
| 1) Propagate a priori particles through measurement model (predicted measurement particles) $y_{k|k-1}^{i} = h(x_{k|k-1}^{i}, u_{k-1}^{i}, v_{k-1}^{i})$ $i \in \{1, 2, ..., N_p\}$ |
| 2. Calculate relative likelihood $q_i$, i.e. $p(y_k | x_{k|k-1})$ from innovation $q_i = \frac{1}{m \cdot \sqrt{\text{det}(\nabla y_{k|k-1})}} \cdot \frac{1}{q_i} \cdot (x_{k|k-1}^{i} - z_k)$, $m$ is dimension of the measurement equation and $V$ is measurement co-variance |
| 3. Normalize $q_i$, $q_i = \frac{q_i}{\sum_{i=1}^{N_p} q_i}$ |
| 4. Resampling for a posteriori estimate |
| 5. A posteriori update $x_{k|k} = \frac{1}{N_p} \sum_{i=1}^{N_p} x_{k|k}^{i}$ |

the a posteriori particles as shown in Table 2. Ris04 algorithm is not much efficient [3] to distribute particles evenly based on their weights. Several other more efficient resampling algorithms are available. In this paper, we focus on multinomial resampling, stratified resampling, residual resampling, and systematic resampling. These resampling algorithms are taken from [5], [6], and [7].

1) MULTINOMIAL RESAMPLING

For multinomial resampling, we first generate an ordered $N_p$ uniform random number in $U(0, 1)$ and use it to select the a priori particles with higher weights based on the index of ordered random numbers. The multinomial resampling algorithm is similar to Ris04 resampling, however, the selection of particles are based on the ordered sequence of random numbers. The multinomial resampling algorithm is given in Table 2 1). In the algorithm, $q^+$ represents the cumulative summation of the likelihood $q_i$, $\text{cumsum}(\cdot)$ represents the cumulative summation and $\text{sort}(\cdot)$ represents the sorting function for creating an ordered sequence. Both $\text{cumsum}(\cdot)$ and $\text{sort}(\cdot)$ are available as a built-in functions in high level programming language for computation.

2) RESIDUAL RESAMPLING

The residual resampling technique works based on modification of normalized likelihoods. The modified likelihood $q^*$ captures the residual from the normalized likelihood $q_i$ based on the floor function of the product of $N_p$ and $q_i$, i.e., $[N_p q_i]$. The modified likelihood is then given as $q^*_i = \frac{N_p q_i - [N_p q_i]}{N_p - \sum_{i=1}^{N_p} [N_p q_i]}$. The modified likelihood can then be used for resampling through the Ris04 or the multinomial resampling. Pseudo code for residual resampling is given in Table 2 2).

3) STRATIFIED RESAMPLING

The stratified resampling algorithm differs from Ris04 and multinomial resampling in that the procedure for generating the sequence of random numbers is different. The algorithm for the stratified resampling algorithm is given in Table 2 3). For $i = 1 : N_p$ we generate the random number as $s_i \sim U\left[\frac{i}{N_p}, \frac{i+1}{N_p}\right]$ and each of these generated numbers $s_i$ are called strata. These strata divides the interval $[0, 1]$ into $N_p$ disjoint sub-intervals $\left(\frac{i}{N_p}, \frac{i+1}{N_p}\right)$ and

$$
\text{sort}(U(0,1)) = \{1, 2, ..., N_p\}
$$

$\text{cumsum}(q^*) = \frac{N_p - \sum_{i=1}^{N_p} [N_p q_i]}{N_p - \sum_{i=1}^{N_p} [N_p q_i]}$ for $r_i \sim U(0,1), i \in 1 : N_p$

for $k = 2, 3, ...$
4) SYSTEMATIC RESAMPLING

Systematic resampling is a modified and computationally more robust algorithm than the stratified resampling algorithm. The random number is generated as 
\[ r \sim U \left( 0, \frac{j - 1}{N_p} \right) \]
and the strata are calculated deterministically using the index as 
\[ s_j = r + j, j \in 1 : N_p. \]

Note that the random number is only generated once in the systematic resampling algorithm. This reduces the computational cost as compared to the other resampling algorithms. Pseudo code for the systematic resampling is given in Table 2 4).

D. SAMPLE IMPOVERISHMENT IN PARTICLE FILTER

The a priori particles are distributed according to the distribution function \( p(x_k | y_{k-1}) \). These particles are resampled using the posterior density function \( p(y_k | x_k) \). In practice, if the high particle density region of the state space of the posterior density \( p(y_k | x_k) \) does not overlap with the high-density state-space region of the prior density \( p(x_k | y_{k-1}) \) then only a few a priori particles with higher weights are selected to become the a posteriori particles during resampling. This process in which there is a significant decrement in the volume of the a priori particles to become a posteriori particles after the resampling is called sample impoverishment. If the sample impoverishment persisted with a fewer number of the a priori particles becoming the a posteriori particles, eventually, all the particles would collapse to the same value. This phenomenon is called a black hole in particle filtering. One way of eradicating this phenomenon is using a large number of particles, however, this increases computational cost. Several solutions are provided to deal with sample impoverishment. Roughening, prior editing, regularized particle filter (RPF), Markov chain Monte Carlo resampling, and auxiliary particle filter (APF) are some of the common methods explained in [3]. This paper will mainly focus on roughening, RPF, and APF.

1) ROUGHENING

Roughening or jittering is a procedure where random noise is added to the a posteriori particles after they are resampled. This increases the volume of the distinct a posteriori particles and the problem of sample impoverishment is improved. Roughening creates diversification in resampled particles which are distributed evenly based on their weights. For the a posteriori estimates roughening can be done as 
\[ x^i_{k|k} \leftarrow x^i_{k|k} + w^i, \]
where \( w_i \) is random noise drawn from a Gaussian distribution with zero-mean and given as 
\[ w_i \sim \left( 0, K \cdot M \cdot N_p \left( \frac{1}{\pi} \right) \right), \]
where \( n_x \) is the number of dimensions of the state space. \( K \) is a scalar tuning factor usually set to 0.2 and \( M \) is the vector containing differences between the maximum and the minimum values of the particles before roughening for the given as,
\[ M = \max \left( x^i_{k|k-1} \right) - \min \left( x^i_{k|k-1} \right). \]
The roughening procedure will diversify the resampled particles by clustering. Roughening also improves statistical moments that we draw from the a posteriori particles.

2) REGULARIZED PARTICLE FILTER (RPF)

The sampling algorithms presented in Table 2 are discrete. However, RPF assumes a probability density function using a continuous distribution. The algorithm for regularized particle filter is taken from [3].

3) AUXILIARY PARTICLE FILTER (APF)

For an auxiliary particle filter (APF) we modify the likelihood obtained in Table 1 by using the following formula,
\[ q_i \leftarrow \frac{(\alpha - 1)q_i + \tilde{q}_i}{\alpha} \quad (4) \]
where \( \alpha \) is a tuning parameter for increasing the diversity in the likelihood. A typical value of \( \alpha \) for APF is 1.1 [3]. During resampling, particles that are outliers having a lower likelihood in the region of the state space are replaced with particles having a higher likelihood. The eradication of these lower weight particles will lose the diversity of distribution of the particles. APF addresses this issue by assigning outliers with the higher likelihood and this is done with Eq. (4). APF is used mostly with highly non-linear systems to address the issues of outliers in the estimation of the posterior distribution.

IV. IMPLEMENTATION OF THE PARTICLE FILTERS

A. INITIALIZATION

The temperature evolution equations of the hydro generator can be written in Differential Algebraic Equations (DAEs) form as
\[ \frac{dx}{dt} = f(x, z, u; \theta) \]
\[ y = h(x, z, u; \theta) \quad (5) \]
where \( x = (T_r, T_s, T_{Fe}), z = (T^c_a, T^\delta_a, T^h_a), u = (U_{I1}, I_1, T_w), \theta = (m_1, m_s, m_{Fe}, R_t, R_s, \tilde{c}_p, \tilde{c}_{cp}, \tilde{c}_{p, Fe}, \tilde{c}_{p, a}, \tilde{m}_a, U_{A1}, U_{A2}). \]
\[ U_{A1, Fe2a}, U_{AFe2a}, \tilde{Q}_{Fe}^c, \tilde{Q}_{Fe}^h, N_{St}^c, N_{St}^h, N_{St}^\delta \]

Out of the three states, \( T_s \) and \( T_{Fe} \) are measured, while it is of interest to estimate the temperature of rotating rotor copper \( T_r \). Similarly, out of three algebraic variables, \( T^c_a \) and \( T^h_a \) are measured, and it is also of interest to estimate air gap temperature \( T^\delta_a \).

The initialization of the particle filter is done as in Table 3.
TABLE 3. Initialization of the particle filter.

| Quantity                        | Symbol | Value          |
|---------------------------------|--------|----------------|
| Process co-variance             | \( \Sigma \) | \( \text{diag}(4, 4, 4) \) |
| Measurement co-variance         | \( \Sigma \) | \( \text{diag}(1, 1) \) |
| Initial state co-variance       | \( \Sigma_{x_0} \) | \( 10 \cdot \Sigma \) |
| Prior distribution of states    | \( p(x_1) \) | \( N(\hat{x}_1, \Sigma_{x_1}) \) |
| Process disturbance             | \( u_x \) | \( N(\hat{w}, \Sigma_{w}) \) |
| Measurement noise               | \( u_e \) | \( N(\hat{v}, \Sigma_{v}) \) |
| Scalar factor for roughening     | \( K \) | 0.2            |
| Diversity tuning parameter for APF | \( \alpha \) | 1.1            |

B. a priori AND a posteriori PARTICLES

Figure 2 shows the a priori and the a posteriori particle spaces and the density function for the rotor copper temperature \( T_r \) at measurement sample \( k = 2 \) using particle filter with Ris04 resampling. The total number of particles is 200. In the figure, we see that without roughening, there is no diversity in the a posteriori particles. We see that most of the a priori particles with the lower weights which will become a posteriori particles after the resampling approach the same value as shown in the figure for particle space of the a posteriori “Ris04” without roughening. This gives rise to bi-modal density function for a posteriori particles for \( T_r \) as shown in the respective density function. On the contrary, when roughening is done with tuning parameter \( K = 0.2 \), there is the diversity in the a posteriori particles as shown in the figure for the particle space of the a posteriori “Ris04-Roughening”.

Figure 3 shows the a posteriori particle space for the rotor copper temperature \( T_r \) at measurement sample \( k = 2 \) using particle filter with multinomial, residual and stratified resamplings, respectively. Similarly, Fig. 4 shows the a posteriori particle space for the rotor copper temperature \( T_r \) at measurement sample \( k = 2 \) using particle filter with systematic, RPF and APF resamplings.
RPF, and APF resamplings, respectively. Furthermore, Fig. 5 shows the \textit{a posteriori} density functions for the rotor copper temperature $T_r$ using particle filter with multinomial, residual, and stratified resamplings.

From Fig. 3 the \textit{a posteriori} particles without roughening for the multinomial resampling, we see that there is the problem of sample impoverishment. The particles with lower weights get clogged into the same value and form a point where lower weights particles are sucked into like in a black hole. Each black hole are in fact the modal value of the \textit{a posteriori} distribution. This creates a poor distribution of the \textit{a posteriori} particles. As the poor distribution propagates through the time series, the error in the estimate increases. One of the solutions to solve the sample impoverishment would be to increase the number of particles. However, this process is computationally demanding. Another solution is to roughen the \textit{a posteriori} particles with a scalar tuning parameter $K = 0.2$. This causes the lower weights \textit{a posteriori} particles to cluster around the modal values preserving the volumes of particles of the \textit{a posteriori} distribution nearly equal to that of the \textit{a priori} and the likelihood. Thus, these clustered particles around the modal values diversify the \textit{a posteriori} distribution to become a unimodal distribution as shown in Fig. 5 for the distribution of the \textit{a posteriori} particles with multinomial roughening. Similar sample impoverishment occurs in the case of the residual resampling, stratified resampling, and systematic resampling as shown in figures Figs. 3 and 4. The sample impoverishment also occurs in the case of the Ris04 resampling as shown in Fig. 2. The sample impoverishment is worst in the case of the stratified resampling. By applying roughening to Ris04 and residual resampling, the \textit{a posteriori} particles are equally weighted so that the \textit{a posteriori} distribution is smooth than other resampling methods.

From Fig. 4 in the case of the \textit{a posteriori} particles with RPF and APF resamplings, we see that the \textit{a posteriori} particles are distributed evenly. There is no problem of sample impoverishment at all after applying these resampling algorithms. Hence, both RPF and APF are good solutions for removing the problem of sample impoverishment. In the figure, we also see that there is no need for the roughening of the particles.

\section*{C. METAL AND AIR TEMPERATURES ESTIMATION}

Figure 6 shows air and metal temperature estimation using particle filter with number of particles $N_p = 200$ using Ris04 resampling. From the figure, $\hat{T}_{r,k|k}$ is the estimated rotor copper temperature where the mean (red), standard deviation (dark gray), and the ensemble of the mean (light gray) are plotted together. Fig. 6 also shows the \textit{a posteriori} estimates of the mean, the standard deviation and the ensemble of the mean plotted together for $T_s$, $T_{Fe}$, $T_{ca}$ and $T_{ha}$. In the figure, $T_{m,s}$, $T_{m,Fe}$, $T_{m,ca}$ and $T_{m,ha}$ are the measurements available for the stator copper temperature, the stator iron temperature, the cooled air temperature and the hot air temperature, respectively. These measurements are used for estimating unmeasured $T_r$ and $T_{\delta}$.  

\begin{figure}[h!]
\centering
\includegraphics[width=\textwidth]{figure5}
\caption{A posteriori density function for the rotor copper temperature $T_r$ at measurement sample $k = 2$ using particle filter with multinomial, residual and stratified resamplings.}
\end{figure}

\begin{figure}[h!]
\centering
\includegraphics[width=\textwidth]{figure6}
\caption{Metal and air temperatures estimation using particle filter with $N_p = 200$ and Ris04 resampling.}
\end{figure}
It is of interest to compare the estimation of unmeasured temperatures $T_r$ and $T_a$ using a different number of particles and different kinds of sampling algorithms. Fig. 7 shows the a posteriori estimate for $T_r$ and $T_a$ with different number of particles using the Ris04, multinomial, and residual resampling. For $\hat{T}_{r,k|k}$ using Ris04 we see that as the number of particles increases from $N_p = 50$ towards $N_p = 1000$, the estimation gets improved. Similar results can be seen from Fig. 7 for both $\hat{T}_{r,k|k}$ and $\hat{T}_{a,k|k}$ using multinomial and residual resamplings. The estimate $\hat{T}_{a,k|k}$ is much better than the estimate $\hat{T}_{r,k|k}$. This adheres that for the same accuracy estimation the realization of the number of particles $N_p$ in case of the metal temperatures is more than in the case of the air temperatures. Similar, results were obtained in the case of a posteriori estimates of $T_r$ and $T_a$ with different number of particles using stratified and systematic resamplings. However, the a posteriori estimates are much better with RPF and APF.

V. PARTICLE FILTER VERSUS KALMAN FILTER
As described in Section III-A Eqs. (1-3) are solved recursively to obtain the state estimation. The analytical solution for Eq. (1) only exists for a few special cases. For instance, if we consider both the process and measurement dynamics as linear functions, and process and measurement noises as Gaussian distribution, then the analytical solution is the optimal linear Kalman filter. However, the optimal Kalman filter can not be derived from the Bayesian formulation for linear dynamical systems with non-Gaussian noises. Kalman filter can also be derived from the least-squares error method as in original KF [8] and it preserves optimality in the case of both Gaussian and non-Gaussian noises for linear systems.

VI. ESTIMATION ACCURACY AND COMPUTATION TIME
It is of interest to see the estimation accuracy $\epsilon$ versus the number of particles $N_p$ for different estimation algorithms. It is also of interest to see the computational effort $\tau$ for each of the estimation algorithms. The estimators were implemented in the Julia language and the computational effort was calculated using Processor: Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz with Installed RAM: 32 GB. All the state estimators’ computational effort is calculated relative to UKF.

The estimated $T_r$ and $T_a$ using both UKF and EnKF with different number of particles are shown in Fig. 8. In the figure, the estimation of $T_r$ and $T_a$ using EnKF shows that as the number of particles in the ensemble increases the estimation improves. The estimation using UKF and EnKF with $N_p$ above 200 gives comparable results. Comparing EnKF from Fig. 8 with PFs with different resamplings from Fig. 7, we see that particle filters with $N_p = 50$ equals EnKF with $N_p = 5$. This shows that for a system with process and measurement noises being Gaussian distribution EnKF with a lower number of particles realization can represent particle filters with a higher number of particles realization.

A. KALMAN FILTER AND ITS VARIANTS
There are several variants of the original KF. As the non-linearity increases in any dynamical system, linear KF fails. For a non-linear system with Gaussian/non-Gaussian noises, state estimation algorithms like Extended Kalman Filter (EKF), UKF [9], and EnKF [10] are employed; UKF and EnKF have several variants based on their performances. However, for a non-linear system with non-Gaussian noises, these variants of the KF can not guarantee a true estimate if the true posterior distribution is multimodal and unsymmetrical around the mean [5]. In reality, the multimodal a posteriori distribution can be approximated by summing up weighted Gaussian for each modal distribution as described in [11]. The Gaussian sum can also be applied to other variants of KF, for instance, Gaussian Sum-UKF (GSUKF) [12]. Several variants of the KF are compared in [13] for a case study to estimate the state of charge in lithium-ion cells. From the paper, it was shown that in addition to the selection of filter algorithm for a particular problem, the tuning of the filter also plays an important role in the estimation accuracy. Similarly, in [14] different variants of EnKF are compared and analyzed for determining the synthetic experiments required to determine the difference in RMSE between the variants.

In this paper, we are more focused on implementing and comparing UKF and EnKF with particle filters. We follow the same notations of our previous work on state estimation for the thermal model of the synchronous generator in [2] where UKF and EnKF algorithms are succinctly defined.

B. ESTIMATION USING UKF AND EnKF
The estimated $T_r$ and $T_a$ using both UKF and EnKF with different number of particles are shown in Fig. 8. In the figure, the estimation of $T_r$ and $T_a$ using EnKF shows that as the number of particles in the ensemble increases the estimation improves. The estimation using UKF and EnKF with $N_p$ above 200 gives comparable results. Comparing EnKF from Fig. 8 with PFs with different resamplings from Fig. 7, we see that particle filters with $N_p = 50$ equals EnKF with $N_p = 5$. This shows that for a system with process and measurement noises being Gaussian distribution EnKF with a lower number of particles realization can represent particle filters with a higher number of particles realization.

FIGURE 7. Metal and air temperatures estimation using particle filter with different number of particles for Ris04, multinomial, and residual resampling. In the figure, $\phi_{k} = 15 - k$ where $k = \{4, 3, 2, 1, 0\}$ is mapped for number of particles $N_p = \{50, 100, 200, 500, 1000\}$ for rotor copper temperature estimate $T_{r,k|k}$. Similarly, $\phi_{k} = 2 - k$ for $T_{a,k|k}$, $\phi_{k} = 15$ and $\phi_{k} = 2$ are introduced for greater visualization of the estimation.
It shows that as the number of particles increases the RMSE of innovation residuals $\epsilon$ decreases. However, the computational time $\tau$ increases. From Table 4 for EnKF we see that as $N_p$ increases, $\epsilon$ converged to 0.9°C. Similar, results can be seen for PF-Ris04, PF-Multinomial, PF-Residual, PF-stratified, and PF-systematic with different $N_p$. Furthermore, from Table 4 for EnKF the computational time $\tau$ for $N_p = 1000$ is 290 times the computational time for UKF whereas to obtain the similar estimation error with $N_p = 1000$ for PF-Ris04, PF-Multinomial, PF-Residual, PF-stratified, and PF-systematic the computational time for these filters are 400 times that of computational time for UKF. However, from Table 4, we see that both PF-RPF and PF-APF suffers from estimation accuracy and computational time.

Figure 9 a) shows computational time $\tau$ versus particle size $N_p$ for EnKF, PF-Ris04 and PF-Multinomial. c) Estimated errors for the stator copper temperature $T_s$ and d) estimated errors for the stator iron temperature $T_{Fe}$.

Table 4. Estimation algorithms, estimation accuracy $\epsilon$ and computational effort $\tau$, PF-Stratified, PF-Systematic, PF-RPF and PF-APF are compared relative to UKF.

| Filters          | RMSE of $\epsilon$ | $\tau$ for $\Delta t = 1$ min |
|------------------|---------------------|-------------------------------|
| UKF              |                     |                               |
| EnKF ($N_p = 5$) | 1.23                | 1.95                          |
| EnKF ($N_p = 20$)| 0.98                | 5.91                          |
| EnKF ($N_p = 50$)| 0.93                | 13.68                         |
| EnKF ($N_p = 200$)| 0.91              | 55.68                         |
| EnKF ($N_p = 500$)| 0.90               | 141.37                        |
| EnKF ($N_p = 1000$)| 0.90             | 290.88                        |
| PF-Ris04 ($N_p = 5$) | 1.72            | 2.47                          |
| PF-Ris04 ($N_p = 20$) | 1.08           | 7.85                          |
| PF-Ris04 ($N_p = 50$) | 0.99            | 19.86                         |
| PF-Ris04 ($N_p = 200$) | 0.96           | 82.34                         |
| PF-Ris04 ($N_p = 500$) | 0.95           | 199.71                        |
| PF-Ris04 ($N_p = 1000$) | 0.96            | 407.46                        |
| PF-Multinomial  ($N_p = 5$) | 1.71          | 2.50                          |
| PF-Multinomial ($N_p = 20$) | 1.12          | 7.86                          |
| PF-Multinomial ($N_p = 50$) | 1.00          | 19.21                         |
| PF-Multinomial ($N_p = 200$) | 0.96          | 82.00                         |
| PF-Multinomial ($N_p = 500$) | 0.96          | 197.48                        |
| PF-Multinomial ($N_p = 1000$) | 0.96          | 403.90                        |
| PF-Residual ($N_p = 5$) | 1.68          | 2.50                          |
| PF-Residual ($N_p = 20$) | 2.066         | 7.88                          |
| PF-Residual ($N_p = 50$) | 1.07          | 19.04                         |
| PF-Residual ($N_p = 200$) | 1.04          | 82.09                         |
| PF-Residual ($N_p = 500$) | 1.02          | 197.23                        |
| PF-Residual ($N_p = 1000$) | 1.02          | 402.40                        |
| PF-Stratified ($N_p = 5$) | 1.71          | 2.49                          |
| PF-Stratified ($N_p = 20$) | 1.11          | 7.85                          |
| PF-Stratified ($N_p = 50$) | 0.99          | 18.85                         |
| PF-Stratified ($N_p = 200$) | 0.960         | 82.86                         |
| PF-Stratified ($N_p = 500$) | 0.952         | 199.07                        |
| PF-Stratified ($N_p = 1000$) | 0.960         | 413.06                        |
| PF-Systematic ($N_p = 5$) | 1.67          | 2.49                          |
| PF-Systematic ($N_p = 20$) | 1.08          | 7.85                          |
| PF-Systematic ($N_p = 50$) | 0.99           | 18.85                         |
| PF-Systematic ($N_p = 200$) | 0.963         | 78.86                         |
| PF-Systematic ($N_p = 500$) | 0.960         | 199.07                        |
| PF-Systematic ($N_p = 1000$) | 0.960         | 413.06                        |
| PF-RPF ($N_p = 5$) | 38.09           | 3.19                          |
| PF-RPF ($N_p = 20$) | 1.76            | 20.20                         |
| PF-RPF ($N_p = 50$) | 1.55            | 80.44                         |
| PF-RPF ($N_p = 200$) | 1.45            | 102.6                        |
| PF-RPF ($N_p = 500$) | 1.42           | 659.65                        |
| PF-RPF ($N_p = 1000$) | 1.43           | 2723.42                       |
| PF-APF ($N_p = 5$) | 11.36           | 2.50                          |
| PF-APF ($N_p = 20$) | 4.14            | 7.87                          |
| PF-APF ($N_p = 50$) | 1.96            | 20.32                         |
| PF-APF ($N_p = 200$) | 1.56            | 82.06                         |
| PF-APF ($N_p = 500$) | 1.44            | 409.80                        |
| PF-APF ($N_p = 1000$) | 1.48           | 200.93                        |

particle filters (Ris04 and Multinomial). Figure 9 b) also shows that as the number of particle size increases there is an exponential decrease in the errors. With the same particle size, EnKF performs better than other particle filters.

Figure 9 c) shows the estimated time series of the innovation error $\epsilon$ for the stator copper temperature $T_s$ by using different estimation filters. In the figure, we see that EnKF $N_p = 1000$ has smaller estimated errors than UKF and PF-Ris04 with $N_p = 1000$. Both UKF and PF-Ris04 with $N_p = 1000$ shows equivalent estimation. Regardless of the computational time, EnKF estimation is better than UKF and PF-Ris04 estimations. The average estimated error difference of EnKF with other filters is about $-0.2°C$.  

[Figure 8: Metal and air temperatures estimation using UKF and EnKF with different number of particles. In the figure, $\phi_1^k = 15 \cdot k$ where $k = \{7, 6, 5, 4, 3, 2, 1, 0\}$ is mapped for number of particles $N_p = \{5, 10, 20, 50, 100, 200, 500, 1000\}$ for rotor copper temperature estimate $\hat{T}_{r,k}^{|k}$. Similarly, $\phi_2^k = 2 \cdot k$ for air gap temperature estimate $\hat{T}_{\delta,a,k}^{|k}$. $\phi_1^k$ and $\phi_2^k$ are introduced for greater visualization of the estimation.]

[Figure 9: a) Computational time $\tau$ versus particle size $N_p$ and b) RMSE of innovation $\epsilon$ versus $N_p$ for EnKF, PF-Ris04 and PF-Multinomial. c) Estimated errors for the stator copper temperature $T_s$ and d) estimated errors for the stator iron temperature $T_{Fe}$.]

[Table 4: Estimation algorithms, estimation accuracy $\epsilon$ and computational effort $\tau$, PF-Stratified, PF-Systematic, PF-RPF and PF-APF are compared relative to UKF.]
Similarly, Fig. 9 c) shows the estimated time series of the innovation error $\epsilon$ for the stator iron temperature $T_{Fe}$ by using different estimation filters. In the figure, we see that EnKF performs better than PF-Ris04 and UKF. The performance of UKF is worst among EnKF and PF-Ris04. The average estimated error difference of UKF with other filters is about $+0.25^\circ C$. Estimation from UKF improves for $t \geq [350]\text{s}$.

VII. CONCLUSION

From Section IV-B, results indicate that all the particle filters except PF-RPF and PF-APF suffer from the sample impoverishment. The sample impoverishment in the sampling algorithms of PFs can be ordered from worst to good as stratified, systematic, multinomial, Ris04, and residual resamplings, respectively. We could not draw conclusions about PF-RPF and PF-APF because both methods suffer from poor estimation accuracy. The issues with PF-RPF and PF-APF will be addressed as future work.

From Section IV-C, results from the PF estimation of the metal and the air temperatures of the generator indicate that the particles realization for the metal temperatures is higher than that of the air temperatures to obtain a similar estimation accuracy. From Section V-B we conclude that regardless of the computational time, UKF estimation is compared with EnKF with the number of particles $N_p = 200$. Furthermore, we also conclude that PFs with $N_p = 50$ gives comparable estimation accuracy with EnKF with $N_p = 5$. This shows that if the posterior distribution of the parameters of the non-linear system is Gaussian distribution EnKF is the sub-optimal filter. This directs us to select PFs in the case of the system’s parameters whose posterior distributions are multimodal.

From Tables 4 we compare the estimation accuracy $\epsilon$ and the computational time $\tau$ with the number of realization of the particles $N_p$ for different PFs and EnKF. The comparison is relative to UKF. Regardless of the computational time for the estimators EnKF with $N_p \geq 100$ converged to estimation error $\approx 0.9$ times the estimation error of UKF. At $N_p = 100$ the computational time for EnKF is about $\approx 30$ times that of UKF. For the realization of particles $N_p \geq 100$ all the estimators’ estimation accuracy converged to a single value representing a sub-optimal solution achieved by the Bayesian inference. This shows that UFK and EnKF are better estimators than PFs for the air-cooled hydro generator system. From Section VI we conclude that keeping the same estimation accuracy of the filters the performance of UKF is worst than EnKF and PFs in the case of the estimation of the stator iron temperature. As a final concluding remark, we choose UKF based on the fair estimation accuracy and the fastest computational time with temperature estimation under standard deviation of $\sigma \approx 0.2^\circ C$.

REFERENCES

[1] T. Oyang, J. K. Noland, G. J. Heggild, and B. Lie, “Online model-based thermal prediction for flexible control of an air-cooled hydrogenerator,” IEEE Trans. Ind. Electron., vol. 66, no. 8, pp. 6311–6320, Aug. 2019.

[2] M. Pandey, “Model fitting and state estimation for thermal model of synchronous generator,” M.S. thesis, Univ. South-Eastern Norway, Notodden, Norway, 2019.

[3] D. Simon, Optimal State Estimation: Kalman, H Infinity, and Nonlinear Approaches. Hoboken, NJ, USA: Wiley, 2006.

[4] N. Gordon, “Beyond the Kalman filter: Particle filters for tracking applications,” in Proc. 7th Int. Conf. Inf. Fusion, 2004, pp. 1–15.

[5] F. Gustafsson, “Particle filter theory and practice with positioning applications,” IEEE Aeros. Electron. Syst. Mag., vol. 25, no. 7, pp. 53–82, Jul. 2010.

[6] L. Tiancheng, B. Miodrag, and D. M. Petar, “Resampling methods for particle filtering: Classification, implementation, and strategies,” Signal Process. Mag., vol. 32, no. 3, pp. 70–86, May 2015.

[7] J. M. Giron-Sierra, Digital Signal Processing With MATLAB Examples, vol. 3. Cham, Switzerland: Springer, 2017.

[8] R. E. Kalman, “A new approach to linear filtering and prediction problems,” J. Basic Eng., vol. 82, pp. 35–45, Oct. 1960.

[9] S. J. Julier and J. K. Uhlmann, “New extension of the Kalman filter to non-linear systems,” in Signal Processing, Sensor Fusion, and Target Recognition VI, vol. 3068. Bellingham, WA, USA: SPIE, 1997, pp. 182–193.

[10] G. Evensen, “The ensemble Kalman filter: Theoretical formulation and practical implementation,” Ocean Dyn., vol. 53, no. 4, pp. 343–367, Nov. 2003.

[11] D. L. Alspach and H. W. Sorenson, “Nonlinear Bayesian estimation using Gaussian sum approximations,” IEEE Trans. Autom. Control, vol. AC-17, no. 4, pp. 439–448, Aug. 1972.

[12] K. K. Kottakki, M. Bhushan, and S. Bhartiya, “An improved Gaussian sum unscented Kalman filter,” IFAC Proc. Volumes, vol. 47, no. 1, pp. 355–362, 2014.

[13] C. Campestrini, T. Heil, S. Kosch, and A. Jossen, “A comparative study and review of different Kalman filters by applying an enhanced validation method,” J. Energy Storage, vol. 8, pp. 142–159, Nov. 2016.

[14] J. Keller, H.-J. H. Franssen, and G. Marguart, “Comparing seven variants of the ensemble Kalman filter: How many synthetic experiments are needed?,” Water Resour. Res., vol. 54, no. 9, pp. 6299–6318, Sep. 2018.

MADHUSUDHAN PANDEY received the M.Sc. degree in electrical power engineering from the University of South-Eastern Norway (USN), Norway, in 2019. Since 2019, he has been a Ph.D. Researcher with TMCC, USN, Porsgrunn, Norway. His research interests include modeling, control, optimization of dynamic systems, and integration of dispatchable renewable energy sources with variable renewable energy sources.

BERNT LIE (Member, IEEE) received the Ph.D. degree in electrical engineering from the Norwegian University of Science and Technology (NTNU), Norway, in 1990. From 1987 to 1991, he was an Assistant and an Associate Professor at NTNU. In 1992, he joined at the University of South-Eastern Norway, where he is currently a Professor of informatics and a Leader with the Telemark Modeling and Control Center. His research interests include control relevant modeling and advanced control, with applications mainly within the process and energy industries.