A Comparison of Filter-based Approaches for Model-based Prognostics

Matthew Daigle
NASA Ames Research Center
Moffett Field, CA 94035
matthew.j.daigle@nasa.gov

Bhaskar Saha
Palo Alto Research Center
Palo Alto, CA 94304
bhaskar.saha@parc.com

Kai Goebel
NASA Ames Research Center
Moffett Field, CA 94035
kai.goebel@nasa.gov

Abstract— **Model-based prognostics approaches use domain knowledge about a system and its failure modes through the use of physics-based models. Model-based prognosis is generally divided into two sequential problems: a joint state-parameter estimation problem, in which, using the model, the health of a system or component is determined based on the observations; and a prediction problem, in which, using the model, the state-parameter distribution is simulated forward in time to compute end of life and remaining useful life. The first problem is typically solved through the use of a state observer, or filter. The choice of filter depends on the assumptions that may be made about the system, and on the desired algorithm performance. In this paper, we review three separate filters for the solution to the first problem: the Daum filter, an exact nonlinear filter; the unscented Kalman filter, which approximates nonlinearities through the use of a deterministic sampling method known as the unscented transform; and the particle filter, which approximates the state distribution using a finite set of discrete, weighted samples, called particles. Using a centrifugal pump as a case study, we conduct a number of simulation-based experiments investigating the performance of the different algorithms as applied to prognostics.**

TABLE OF CONTENTS

1. INTRODUCTION

Model-based prognostics approaches employ domain knowledge about a system, its components, and how they fail through the use of physics-based models, derived from first principles that capture the underlying physical phenomena. Model-based prognosis is generally divided into two sequential problems: (i) a joint state-parameter estimation problem, in which, using the model, the health of a system is determined based on the observations, and (ii) a prediction problem, in which, using the model, the state-parameter distribution is simulated forward in time to compute end of life (EOL) and remaining useful life (RUL). The first problem is typically solved through the use of a state observer, or filter. The choice of filter depends on the assumptions that may be made about the system, and on the desired algorithm performance and other characteristics.

The most well-known filter is the Kalman filter, which is an exact filter for linear systems with additive Gaussian noise, i.e., it is the optimal filter for these types of systems. Kalman filters have been used for prognostics problems in [1, 2]. In many cases, however, especially with prognostics, systems are nonlinear, and the Kalman filter cannot be used. For special classes of nonlinear dynamics, exact filters are available such as the Beneš filter [3] and the Daum filter [4] along with its variants. Such filters are advocated for prognostics applications in [5], although the classes of dynamics they support are quite restrictive in practice. Approximate filters may also be used, such as the extended Kalman filter and the unscented Kalman filter [6], which are restricted also to additive Gaussian noise. Another nonlinear filter is the particle filter, which approximates the state distribution using a finite set of discrete, weighted samples, called particles [7]. The particle filter does not restrict the dynamics or the noise in any way, but suffers from a high computational complexity. Still, particle filters, due to their general applicability and ease of implementation, have been a popular choice for nonlinear estimation within prognostics, e.g., [8–10].

In this paper, we develop a general model-based prognostics methodology that uses filters for the joint state-parameter estimation step. We review three separate nonlinear filters: (*i*) the Daum filter, (*ii*) the unscented Kalman filter, and (*iii*) the particle filter. Each of these approaches differs in the formulation of the underlying model, the assumptions that they make, their representation of the state-parameter distribution, the computational complexity, and the performance that may be achieved. Using a centrifugal pump as a case study, we conduct a number of simulation-based experiments investigating the performance trade offs of the different prognostic algorithms. Since the Daum filter does not support the class of dynamics that the centrifugal pump model falls in, we perform experiments only using the unscented Kalman filter and the particle filter. Algorithm performance is measured using established prognostics metrics [11].

The paper is organized as follows. Section 2 formally defines the prognostics problem and describes the model-based prognostics architecture. Section 3 describes different estimation methods and develops a flexible variance control scheme. Section 4 discusses the prediction methodology. Section 5 describes the case study and provides results from a number of simulation-based experiments and evaluates the approach. Section 6 concludes the paper.

2. MODEL-BASED PROGNOSTICS

The problem of prognostics is to predict the EOL and/or the RUL of a component, subsystem, or system. We assume the

978-1-4577-0557-1/12/$26.00 ©2012 IEEE.
[1] IEEEAC Paper #1121, Version 3, Updated 20/01/2011.

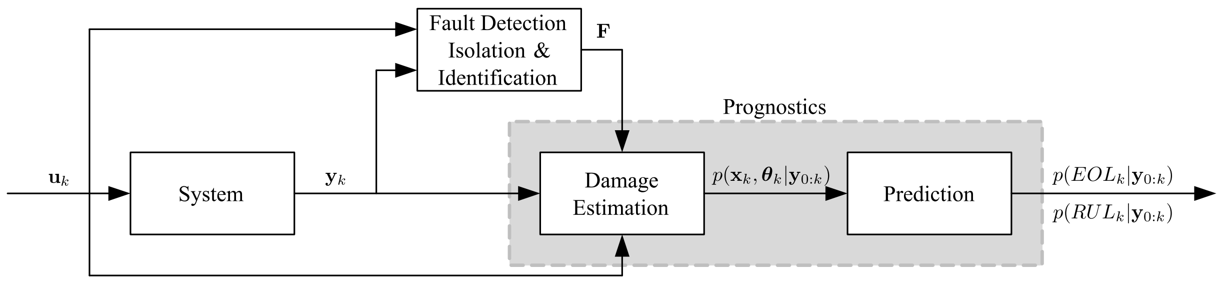
Fault Detection Isolation & Identification
$\mathbf{F}$
Prognostics
$\mathbf{u}_k$
System
$\mathbf{y}_k$
Damage Estimation
$p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k})$
Prediction
$p(EOL_k | \mathbf{y}_{0:k})$
$p(RUL_k | \mathbf{y}_{0:k})$

Figure 1. Prognostics architecture.

system model may be generally defined as

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{v}(t))$$
$$\mathbf{y}(t) = \mathbf{h}(t, \mathbf{x}(t), \boldsymbol{\theta}(t), \mathbf{u}(t), \mathbf{n}(t)),$$

where $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{\theta}(t) \in \mathbb{R}^{n_\theta}$ is the unknown parameter vector, $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ is the input vector, $\mathbf{v}(t) \in \mathbb{R}^{n_v}$ is the process noise vector, $\mathbf{f}$ is the state equation, $\mathbf{y}(t) \in \mathbb{R}^{n_y}$ is the output vector, $\mathbf{n}(t) \in \mathbb{R}^{n_n}$ is the measurement noise vector, and $\mathbf{h}$ is the output equation. This is a general nonlinear model with no restrictions on the functional forms of $\mathbf{f}$ or $\mathbf{h}$, or on how the noise terms are coupled with the states and parameters.

In prognostics, we are interested in when the performance of a system lies outside some desired region of acceptable behavior. Outside this region, we say that the system has failed. The desired performance is expressed through a set of c constraints, $\mathcal{C} = \{C_i\}_{i=1}^c$, where C_i is a function

$$C_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_\theta} \to \mathbb{B}$$

that maps a given point in the joint state-parameter space, $(\mathbf{x}(t), \boldsymbol{\theta}(t))$, to the Boolean domain $\mathbb{B} \triangleq [0, 1]$, where $C_i(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 1$ if the constraint is satisfied. If $C_i(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 0$, then the constraint is not satisfied and the system has failed.

These individual constraints may be combined into a single threshold function T_{EOL}, where

$$T_{EOL} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_\theta} \to \mathbb{B},$$

defined as

$$T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t)) = \begin{cases} 1, & 0 \in \{C_i(\mathbf{x}(t), \boldsymbol{\theta}(t))\}_{i=1}^c \\ 0, & \text{otherwise.} \end{cases}.$$

That is, T_{EOL} evaluates to 1, i.e., the system has failed, when any of the constraints are violated.

At some point in time, t_P, the system is at $(\mathbf{x}(t_P), \boldsymbol{\theta}(t_P))$ and we are interested in predicting the time point t at which this state will evolve to $(\mathbf{x}(t), \boldsymbol{\theta}(t))$ such that $T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 1$. Using T_{EOL}, we formally define EOL with

$$EOL(t_P) \triangleq \inf\{t \in \mathbb{R} : t \geq t_P \wedge T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 1\},$$

i.e., EOL is the earliest time point at which T_{EOL} is met. RUL is expressed using EOL as

$$RUL(t_P) \triangleq EOL(t_P) - t_P.$$

The model-based prognostics architecture is shown in Fig. 1. In discrete time k, the system is provided with inputs $\mathbf{u}_k$ and provides measured outputs $\mathbf{y}_k$. The damage estimation module uses this information, along with the system model, to compute an estimate $p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k})$. While the damage estimation module may be implemented in various ways, in this paper, we focus on filtering solutions to this problem. The prediction module uses the joint state-parameter distribution and the system model, along with hypothesized future inputs, to compute EOL and RUL as probability distributions $p(EOL_{k_P} | \mathbf{y}_{0:k_P})$ and $p(RUL_{k_P} | \mathbf{y}_{0:k_P})$ at given prediction times k_P. A fault detection, isolation, and identification (FDII) module may be used in parallel to determine which damage mechanisms are active, represented as a fault set $\mathbf{F}$, which may enable the damage estimation module to limit the dimension of the joint state-parameter space that must be estimated. In this paper, we assume such a module is absent.

3. Damage Estimation

In model-based prognostics, damage estimation reduces to joint state-parameter estimation, i.e., computation of $p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k})$. A common solution to this problem is to use a filter, where the unknown parameters simply augment the state vector. Treating parameters as states in most cases makes the system nonlinear, if not already so. In any case, most models for prognostics are nonlinear because damage progression models are rarely linear. Therefore, we focus here on nonlinear filters. Specifically, we consider the Daum filter [4], which is an exact nonlinear filter for a specific class of nonlinear functions, the unscented Kalman filter (UKF) [6, 12], which is an approximate nonlinear filter that uses deterministic sampling, and the particle filter (PF) [7], which is an approximate nonlinear filter that uses stochastic sampling.

Daum Filter

With the Daum filter, the system model is defined by

$$\dot{\mathbf{x}}(t) = \mathbf{f}(t, \mathbf{x}(t), \mathbf{u}(t)) + \mathbf{G}(t)\frac{d\mathbf{v}}{dt} \tag{1}$$
$$\mathbf{y}(t) = \mathbf{H}(t)\mathbf{x}(t) + \mathbf{n}(t), \tag{2}$$

where $\mathbf{G}(t)$ is the process noise matrix (with the covariance of $\mathbf{v}$ being $\mathbf{I}$), and $\mathbf{H}(t)$ is the observation matrix. The sensor noise has covariance $\mathbf{R}$. Note the following restrictions on the system model. First, the process noise is restricted to be additive and Brownian. Second, the output equation is assumed to be linear, and the sensor noise is assumed to be additive and Gaussian.

The system model is restricted further to satisfy certain assumptions. First, there must exist a function $\boldsymbol{\psi}(\mathbf{x}, t)$

that satisfies the following Fokker-Planck partial differential equation (PDE):

$$\frac{\partial \psi}{\partial t} = -\frac{\partial \psi}{\partial \mathbf{x}}\mathbf{f} - \psi \text{tr}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right) + \frac{1}{2}\text{tr}\left(\mathbf{Q}\frac{\partial^2 \psi}{\partial \mathbf{x}^2}\right), \quad (3)$$

where $\mathbf{Q} = \mathbf{G}\mathbf{G}^T$. Further, there must exist $\mathbf{A}$, $\mathbf{b}$, $\mathbf{c}$, $\mathbf{D}$, and $\mathbf{E}$ such that

$$\text{tr}\left(\frac{\partial \mathbf{f}}{\partial \mathbf{x}}\right) + (s/2)\mathbf{r}\mathbf{Q}\mathbf{r}^T = \mathbf{x}^T\mathbf{A}\mathbf{x} + \mathbf{b}^T\mathbf{x} + \mathbf{c} \quad (4)$$

$$\mathbf{f} - s\mathbf{Q}\mathbf{r}^T = \mathbf{D}\mathbf{x} + \mathbf{E}, \quad (5)$$

where $0 < s < 1$, and $\mathbf{r}(\mathbf{x}, t) = \partial/\partial\mathbf{x}(\ln \psi(\mathbf{x}, t))$.

If these conditions hold, in addition to some additional assumptions (see [4]), then we have

$$p(\mathbf{x}_k|\mathbf{y}_{0:k}) = \psi(\mathbf{x}_k, k)\cdot \exp(-\frac{1}{2}(\mathbf{x}_k - \mathbf{m}_k)^T\mathbf{P}_k^{-1}(\mathbf{x}_k - \mathbf{m}_k)). \quad (6)$$

Here, $\mathbf{m}$ and $\mathbf{P}$ are the filter variables and are analogous to mean and covariance. The filter computes them using

$$\mathbf{m}_k = \bar{\mathbf{m}}_k + \mathbf{P}_k\mathbf{H}_k^T\mathbf{R}_k^{-1}(\mathbf{y_k} - \mathbf{H_k}\bar{\mathbf{m}}_k) \quad (7)$$

$$\mathbf{P}_k = \bar{\mathbf{P}}_k - \bar{\mathbf{P}}_k\mathbf{H}_k^T(\mathbf{H}_k\bar{\mathbf{P}}_k\mathbf{H}_k^T + \mathbf{R}_k)^{-1}\mathbf{H}_k\bar{\mathbf{P}}_k. \quad (8)$$

The auxiliary variables $\bar{\mathbf{m}}_k$ and $\bar{\mathbf{P}}_k$ are found by solving the following equations over t_{k-1} to t_k with initial conditions $\bar{\mathbf{m}}_k = \mathbf{m}_{k-1}$ and $\bar{\mathbf{P}}_k = \mathbf{P}_{k-1}$:

$$d\bar{\mathbf{m}}(t)/dt = 2(s-1)\bar{\mathbf{P}}(t)\mathbf{A}(t)\bar{\mathbf{m}}(t) + \mathbf{D}(t)\bar{\mathbf{m}}(t) + (s-1)\bar{\mathbf{P}}(t)\mathbf{b}(t) + \mathbf{E}(t) \quad (9)$$

$$d\bar{\mathbf{P}}(t)/dt = 2(s-1)\bar{\mathbf{P}}(t)\mathbf{A}(t)\bar{\mathbf{P}}(t) + \mathbf{D}(t)\bar{\mathbf{P}}(t) + \bar{\mathbf{P}}(t)\mathbf{D}^T(t) + \mathbf{Q}(t). \quad (10)$$

Both the Kalman and Beneš filters may be derived from the Daum filter when the assumptions for these filters are made, as shown in [4]. The main advantage of the approach is that, like the Kalman filter for linear Gaussian systems, it is an exact filter for the class of nonlinear dynamics that it covers. Further, it has computational complexity on the order of the Kalman filter. The main disadvantage is that the class of nonlinear dynamics that it covers is quite restrictive for practical application, as we will see in Section 5. In addition, it covers only additive process and sensor noise that are of Brownian and Gaussian distributions, respectively. This, however, is acceptable for many real systems. Another disadvantage is that the Daum filter requires the (offline) analytical solution of a PDE ($\psi(\mathbf{x}, t)$) to both check that the model in question satisfies the constraints of the filter, and for use within the filter equations. It is quite possible that a solution to this PDE does not exist or cannot be found with the available tools.

Unscented Kalman Filter

When exact nonlinear filters are not applicable, approximate methods are required. The most familiar is the extended Kalman filter (EKF), which approximates the Kalman filter by linearizing the dynamics around the operating point. However, when the model dynamics are highly nonlinear, the performance of the EKF suffers [6]. Further, the EKF requires the derivation of Jacobians, which may be difficult to derive analytically, or expensive to compute online, and in some cases do not exist (e.g., when there are discontinuities). In the most basic sense, the EKF linearizes the model at each step and then applies the Kalman filter equations. The unscented Kalman filter, instead of approximating the nonlinearity, approximates the state distribution [6, 12]. This procedure maintains the nonlinear functions exactly, eliminating the need to calculate Jacobians, and thereby providing an easier implementation framework.

The UKF approximates a distribution using the unscented transform (UT). The UT takes a random variable $\mathbf{x} \in \mathbb{R}^{\mathbf{n_x}}$, with mean $\bar{\mathbf{x}}$ and covariance $\mathbf{P_{xx}}$, which is related to a second random variable $\mathbf{y}$ by some nonlinear function $\mathbf{y} = \mathbf{g}(\mathbf{x})$, and computes the mean $\bar{\mathbf{y}}$ and covariance $\mathbf{P_{yy}}$ using a (small) set of *deterministically* selected weighted samples, called *sigma points* [6]. $\boldsymbol{\mathcal{X}}^i$ denotes the ith sigma point from $\mathbf{x}$ and w^i denotes its weight. The sigma points are always chosen such that the mean and covariance match those of the original distribution, $\bar{\mathbf{x}}$ and $\mathbf{P_{xx}}$. Each sigma point is passed through $\mathbf{g}$ to obtain new sigma points $\boldsymbol{\mathcal{Y}}$, i.e.,

$$\boldsymbol{\mathcal{Y}}^i = \mathbf{g}(\boldsymbol{\mathcal{X}}^\mathbf{i}) \quad (11)$$

with mean and covariance calculated as

$$\bar{\mathbf{y}} = \sum_i w^i \boldsymbol{\mathcal{Y}}^i \quad (12)$$

$$\mathbf{P}_{yy} = \sum_i w^i(\boldsymbol{\mathcal{Y}}^i - \bar{\mathbf{y}})(\boldsymbol{\mathcal{Y}}^i - \bar{\mathbf{y}})^T. \quad (13)$$

Several methods exist for selecting sigma points, but we describe here only the commonly used symmetric unscented transform, in which $2n_x + 1$ sigma points are selected symmetrically about the mean in the following way [12]:

$$w^i = \begin{cases} \dfrac{\kappa}{(n_x+\kappa)}, & i = 0 \\ \dfrac{1}{2(n_x+\kappa)}, & i = 1, \ldots, 2n_x \end{cases} \quad (14)$$

$$\boldsymbol{\mathcal{X}}^i = \begin{cases} \bar{\mathbf{x}}, & i = 0 \\ \bar{\mathbf{x}} + \left(\sqrt{(n_x+\kappa)\mathbf{P}_{xx}}\right)^i, & i = 1, \ldots, n_x \\ \bar{\mathbf{x}} - \left(\sqrt{(n_x+\kappa)\mathbf{P}_{xx}}\right)^i, & i = n_x+1, \ldots, 2n_x \end{cases}, \quad (15)$$

where $\left(\sqrt{(n_x+\kappa)\mathbf{P}_{xx}}\right)^i$ refers to the ith column of the matrix square root of $(n_x + \kappa)\mathbf{P}_{xx}$ (e.g., computed using the Cholesky decomposition). The number κ is a free parameter that can be used to tune the higher order moments of the distribution. Note that the sigma point weights do not directly represent probabilities, so are not restricted to the interval $[0, 1]$. If $\mathbf{x}$ is assumed Gaussian, then selecting $\kappa = 3 - n_x$ is recommended [6]. A smaller value of κ will bring the sigma points closer together than a larger value.

The UKF assumes the general nonlinear form of the state and output equations, but is restricted to additive Gaussian noise. It follows the same basic form as the Kalman and extended Kalman filters, modified to use the sigma points. First, n_s sigma points $\hat{\boldsymbol{\mathcal{X}}}_{k-1|k-1}$ are derived from the current mean $\hat{\mathbf{x}}_{k-1|k-1}$ and covariance estimates $\mathbf{P}_{k-1|k-1}$ using the sigma point selection algorithm of choice. The prediction step is:

$$\hat{\boldsymbol{\mathcal{X}}}^i_{k|k-1} = \mathbf{f}(\hat{\boldsymbol{\mathcal{X}}}^i_{k-1|k-1}, \mathbf{u}_{k-1}), i = 1, \ldots, n_s \tag{16}$$

$$\hat{\boldsymbol{\mathcal{Y}}}^i_{k|k-1} = \mathbf{h}(\hat{\boldsymbol{\mathcal{X}}}^i_{k|k-1}), i = 1, \ldots, n_s \tag{17}$$

$$\hat{\mathbf{x}}_{k|k-1} = \sum_i^{n_s} w^i \boldsymbol{\mathcal{X}}^i_{k|k-1} \tag{18}$$

$$\hat{\mathbf{y}}_{k|k-1} = \sum_i^{n_s} w^i \boldsymbol{\mathcal{Y}}^i_{k|k-1} \tag{19}$$

$$\mathbf{P_{k|k-1}} = \mathbf{Q} + \sum_i^{n_s} w^i (\boldsymbol{\mathcal{X}}^i_{k|k-1} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{X}}^i_{k|k-1} - \hat{\mathbf{x}}_{k|k-1})^T, \tag{20}$$

where $\mathbf{Q}$ is the process noise covariance matrix. The update step is:

$$\mathbf{P}_{yy} = \mathbf{R} + \sum_i^{n_s} w^i (\boldsymbol{\mathcal{Y}}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1})(\boldsymbol{\mathcal{Y}}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1})^T \tag{21}$$

$$\mathbf{P}_{xy} = \sum_i^{n_s} w^i (\boldsymbol{\mathcal{X}}^i_{k|k-1} - \hat{\mathbf{x}}_{k|k-1})(\boldsymbol{\mathcal{Y}}^i_{k|k-1} - \hat{\mathbf{y}}_{k|k-1})^T \tag{22}$$

$$\mathbf{K}_k = \mathbf{P}_{xy}\mathbf{P}_{yy}^{-1} \tag{23}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k(\mathbf{y}_k - \hat{\mathbf{y}}_{k|k-1}) \tag{24}$$

$$\mathbf{P}_{k|k} = \mathbf{P}_{k|k-1} - \mathbf{K}_k \mathbf{P}_{yy} \mathbf{K}_k^T, \tag{25}$$

where $\mathbf{R}$ is the sensor noise covariance matrix.

Particle Filter

The particle filter is the most general nonlinear filter, as it may be directly applied to nonlinear systems with non-Gaussian noise terms [7]. In particle filters, the state distribution is approximated by a set of discrete weighted samples, called *particles*. The particle approximation to the state distribution is given by

$$\{\mathbf{x}^i_k, w^i_k\}_{i=1}^N, \tag{26}$$

where N denotes the number of particles, and for particle i, $\mathbf{x}^i_k$ denotes the state vector estimate and w^i_k denotes the weight. The posterior density is approximated by

$$p(\mathbf{x}_k|\mathbf{y}_{0:k}) \approx \sum_{i=1}^N w^i_k \delta_{\mathbf{x}^i_k}(d\mathbf{x}_k), \tag{27}$$

where $\delta_{\mathbf{x}^i_k}(d\mathbf{x}_k)$ denotes the Dirac delta function located at $\mathbf{x}^i_k$.

The PF has many variants. Here, we describe the sampling importance resampling (SIR) particle filter, using systematic resampling [13]. The pseudocode for a single step of the SIR filter is shown as Algorithm 1. Each particle is propagated forward to time k by sampling new states using the model. The particle weight is assigned using $\mathbf{y}_k$. The weights are then normalized, followed by the resampling step [7].

Algorithm 1 SIR Filter

```
Inputs: $\{\mathbf{x}^i_{k-1}, w^i_{k-1}\}_{i=1}^N, \mathbf{u}_{k-1:k}, \mathbf{y}_k$
Outputs: $\{\mathbf{x}^i_k, w^i_k\}_{i=1}^N$
for $i = 1$ to $N$ do
    $\mathbf{x}^i_k \sim p(\mathbf{x}_k|\mathbf{x}^i_{k-1}, \mathbf{u}_{k-1})$
    $w^i_k \leftarrow p(\mathbf{y}_k|\mathbf{x}^i_k, \mathbf{u}_k)$
end for
$W \leftarrow \sum_{i=1}^N w^i_k$
for $i = 1$ to $N$ do
    $w^i_k \leftarrow w^i_k / W$
end for
$\{\mathbf{x}^i_k, w^i_k\}_{i=1}^N \leftarrow \texttt{Resample}(\{\mathbf{x}^i_k, w^i_k\}_{i=1}^N)$
```

Variance Control

In each of these algorithms, only state estimation is described. Joint state-parameter estimation can be accomplished by augmenting the state vector with the unknown parameters. To do this, some type of evolution must be assigned to the parameters. The typical solution is to use a random walk, i.e., for parameter θ, $\theta_k = \theta_{k-1} + \xi_{k-1}$, where ξ_{k-1} is sampled from some distribution (e.g., zero-mean Gaussian). In the PF, this is represented explicitly. In the Daum filter and the UKF, this is represented by setting the corresponding diagonal element of the process noise matrix ($\mathbf{G}$ for the Daum filter and $\mathbf{Q}$ for the UKF) to a nonzero value. In this way, the parameter estimates become time-varying and are modified by the filter using the measured outputs.

The selected variance of the random walk noise determines both the rate of parameter estimation convergence and the estimation performance once convergence is achieved. A large random walk variance gives fast convergence but tracking with too wide a variance, but too small a random walk variance will give very slow convergence, if at all, but, once achieved, tracking will proceed with a small variance. In prognostics, since wear parameter values are generally unknown, except possibly for the order of magnitude, it becomes very difficult to tune the variance value to get the best performance. This has resulted in several approaches that attempt to tune this variance value online, e.g., [8,10,14]. We generalize our previous approach to this problem described in [10], which was developed only for particle filters, and extend it to work with other nonlinear filters.

In this approach, the algorithm tries to control the variance of the hidden wear parameter estimate to a user-specified range by modifying the random walk noise variance. Since the random walk noise is artificial, we should reduce it as much as possible, because this uncertainty propagates into the EOL predictions. So, controlling this uncertainty also helps to control the uncertainty of the EOL prediction.

The algorithm for the adaptation of the random walk variance vector, $\mathbf{v}_\xi$, is given as Algorithm 2. We assume that the distributions that the elements of $\boldsymbol{\xi}$ are drawn from can be specified using a variance value, and that the variance values are tuned initially based on the maximum expected wear rates, e.g., if the pump is expected to fail no earlier than 100 hours, then this corresponds to particular maximum wear rate values. Basically, the algorithm computes the actual spread of a parameter estimate from the filter estimate, and computes the error with the desired level of spread. The variance value for that parameter is then modified to reduce the error. We use a *relative* measure of spread, such as relative standard

Algorithm 2 $\mathbf{v}_\xi$ Adaptation

```
Inputs: $p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k})$
State: $\mathbf{v}_{\xi,k-1}, \mathbf{l} \leftarrow \mathbf{1}$
Outputs: $\mathbf{v}_{\xi,k}$
for all $j \in \{1, 2, \ldots, n_\theta\}$ do
    $v_j \leftarrow \texttt{RelativeSpread}(p(\boldsymbol{\theta}_k(j)|\mathbf{y}_{0:k}))$
    if $v_j < \mathbf{t}_j(\mathbf{s}(\mathbf{j}))$ then
        $\mathbf{s}(j) \leftarrow \mathbf{s}(j) + 1$
    end if
    $\mathbf{v}_{\xi,k}(j) \leftarrow \boldsymbol{\xi}_{k-1}(j)\left(1 + \mathbf{P}_j(\mathbf{s}(j))\frac{v_j - \mathbf{v}_j^*(\mathbf{s}(j))}{\mathbf{v}_j^*(\mathbf{s}(j))}\right)$
end for
$\mathbf{v}_{\xi,k-1} \leftarrow \mathbf{v}_{\xi,k}$
```

deviation or relative median absolute deviation, which can be treated equally for any wear parameter value.

The adaptation proceeds in multiple stages, maintained with an $\mathbf{s}_j$ variable for each parameter (with j referring to the parameter index), with the number of stages specified by S_j. The $\mathbf{s}_j$ values are initialized to 1. Each stage is specified using three variables, (*i*) a lower threshold that, once crossed, signals that the next stage should be entered, (*ii*) the desired spread value for the stage, and (*iii*) a proportional gain term used to control the degree of adaptation during that stage. For each parameter, the threshold vector $\mathbf{t}_j$, the desired spread vector $\mathbf{v}_j^*$, and the proportional gain vector $\mathbf{P}_j$ are of size S_j.

The algorithm works as follows. For each parameter, indexed by j, the current relative spread is computed as v_j using the metric of choice. If this value is below the threshold value for the the current stage, $\mathbf{t}_j(\mathbf{s}(j))$, then the stage number is increased. Then the new random walk variance $\mathbf{v}_{\xi,k}(j)$ is computed. The error between the the actual and the desired spread value for the current stage, $v_j - \mathbf{v}_j^*(\mathbf{s}(j))$, is normalized by $\mathbf{v}_j^*(\mathbf{s}(j))$. This normalized error is then multiplied by the proportional gain term for the current stage, $\mathbf{P}_j(\mathbf{s}(j))$, and the corresponding variance $\mathbf{v}_{\xi,k-1}(j)$ is increased or decreased by that percentage to compute the new variance value $\mathbf{v}_{\xi,k}(j)$.

Because there is some inertia to the process of v_j changing in response to a new value of $\mathbf{v}_{\xi,k}(j)$, the gains $\mathbf{P}_j$ cannot be too large, otherwise v_j will not converge to the desired value, instead, it will continually shrink and expand. So, tuning of these gains and the other algorithm parameters is necessary.

4. PREDICTION

Prediction is initiated at a given time k_P. Using the current joint state-parameter estimate, $p(\mathbf{x}_{k_P}, \boldsymbol{\theta}_{k_P}|\mathbf{y}_{0:k_P})$, which represents the most up-to-date knowledge of the system at time k_P, the goal is to compute $p(EOL_{k_P}|\mathbf{y}_{0:k_P})$ and $p(RUL_{k_P}|\mathbf{y}_{0:k_P})$.

The representation of $p(\mathbf{x}_{k_P}, \boldsymbol{\theta}_{k_P}|\mathbf{y}_{0:k_P})$ is dictated by the selected filter for damage estimation. In the case of the UKF and PF, the distribution is represented by a finite set of weighted samples. If the distribution is given in an analytical form or represented by sufficient statistics (e.g., the mean and covariance matrix), then generally the distribution cannot be analytically propagated to EOL due to the nonlinearity of the model, therefore, we must sample from these representations. Either random sampling of a sufficient number of samples can be performed, or the unscented transform can be used to

Algorithm 3 EOL Prediction

```
Inputs: $\{(\mathbf{x}_{k_P}^i, \boldsymbol{\theta}_{k_P}^i), w_{k_P}^i\}_{i=1}^N$
Outputs: $\{EOL_{k_P}^i, w_{k_P}^i\}_{i=1}^N$
for $i = 1$ to $N$ do
    $k \leftarrow k_P$
    $\mathbf{x}_k^i \leftarrow \mathbf{x}_{k_P}^i$
    $\boldsymbol{\theta}_k^i \leftarrow \boldsymbol{\theta}_{k_P}^i$
    while $T_{EOL}(\mathbf{x}_k^i, \boldsymbol{\theta}_k^i) = 0$ do
        Predict $\hat{\mathbf{u}}_k$
        $\boldsymbol{\theta}_{k+1}^i \sim p(\boldsymbol{\theta}_{k+1}|\boldsymbol{\theta}_k^i)$
        $\mathbf{x}_{k+1}^i \sim p(\mathbf{x}_{k+1}|\mathbf{x}_k^i, \boldsymbol{\theta}_k^i, \hat{\mathbf{u}}_k)$
        $k \leftarrow k + 1$
        $\mathbf{x}_k^i \leftarrow \mathbf{x}_{k+1}^i$
        $\boldsymbol{\theta}_k^i \leftarrow \boldsymbol{\theta}_{k+1}^i$
    end while
    $EOL_{k_P}^i \leftarrow k$
end for
```

deterministically select a small number of samples [15].

Given the finite set of N samples, $\{(\mathbf{x}_{k_P}^i, \boldsymbol{\theta}_{k_P}^i), w_{k_P}^i\}_{i=1}^N$, we simply propagate each sample i out to EOL and use the original sample weight for the weight of that EOL prediction. Each sample is simulated forward to EOL to obtain the complete EOL distribution. The pseudocode for the prediction procedure is given as Algorithm 3 [16]. Each sample i is propagated forward until $T_{EOL}(\mathbf{x}_k^i, \boldsymbol{\theta}_k^i)$ evaluates to 1; at this point EOL has been reached for this sample. In general, prediction requires hypothesizing future inputs of the system, $\hat{\mathbf{u}}_k$, but in this paper, we assume future inputs are known.

5. CASE STUDY

We adopt a centrifugal pump as a simulation-based case study. In this section, we review the pump model (originally described in [10]) and present a number of prognostics experiments. The pump model does not satisfy the requirements for the Daum filter, e.g., it has square roots and states appearing in the denominator, therefore we apply only the unscented Kalman filter and the particle filter. We compare the results of these algorithms on the case study. In both cases, we apply the variance control algorithm using relative standard deviation (RSD) as the measure of spread.

Pump Modeling

A schematic of a typical centrifugal pump is shown in Fig. 2. Fluid enters the inlet, and the rotation of the impeller, driven by a motor, forces fluid through the outlet. Radial and thrust bearings help to minimize friction along the shaft, and are lubricated by oil in the bearing housing. A seal prevents fluid flow into the bearing housing, and wear rings help to minimize internal leakage from the outlet to the inlet side of the impeller.

The state of the pump is given by

$$\mathbf{x}(t) = [\omega(t) \quad T_t(t) \quad T_r(t) \quad T_o(t)]^T, \tag{28}$$

where $\omega(t)$ is the rotational velocity of the pump, $T_t(t)$ is the thrust bearing temperature, $T_r(t)$ is the radial bearing temperature, and $T_o(t)$ is the oil temperature.

The rotational velocity of the pump is described using a

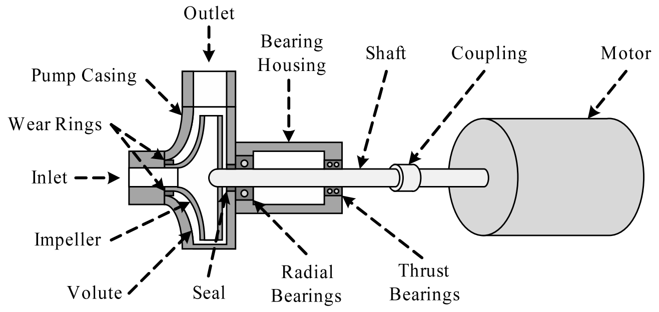

Figure 2. Centrifugal pump.

torque balance,

$$\dot{\omega} = \frac{1}{J}\left(\tau_e(t) - r\omega(t) - \tau_L(t)\right), \tag{29}$$

where J is the lumped motor/pump inertia, τ_e is the electromagnetic torque provided by the motor, r is the lumped friction parameter, and τ_L is the load torque. In an induction motor, a torque is produced only when there is a difference, called *slip*, between the synchronous speed of the supply voltage, ω_s and the mechanical rotation, ω, defined as

$$s = \frac{\omega_s - \omega}{\omega_s}. \tag{30}$$

Torque τ_e is expressed from an equivalent circuit representation for a three-phase induction motor, based on rotor and stator resistances and inductances, and the slip s [17]:

$$\tau_e = \frac{npR_2}{s\omega_s}\frac{V_{rms}^2}{(R_1 + R_2/s)^2 + (\omega_s L_1 + \omega_s L_2)^2}, \tag{31}$$

where R_1 is the stator resistance, L_1 is the stator inductance, R_2 is the rotor resistance, L_2 is the rotor inductance, n is the number of phases, and p is the number of magnetic pole pairs. Rotor speed is controlled by changing the input frequency ω_s.

The load torque τ_L is a function of the flow rate through the pump and the impeller rotational velocity [18, 19]:

$$\tau_L = a_0\omega^2 + a_1\omega Q - a_2 Q^2, \tag{32}$$

where Q is the flow, and a_0, a_1, and a_2 are coefficients derived from the pump geometry [19].

The rotation of the impeller creates a pressure difference from the inlet to the outlet of the pump, which drives the pump flow, Q. Pump pressure is computed as

$$p_p = A\omega^2 + b_1\omega Q - b_2 Q^2, \tag{33}$$

where A is the impeller area, and b_1 and b_2 are coefficients derived from the pump geometry. Flow through the impeller, Q_i, is computed using the pressure differences:

$$Q_i = c\sqrt{|p_s + p_p - p_d|}sign(p_s + p_p - p_d), \tag{34}$$

where c is a flow coefficient, p_s is the suction pressure, and p_d is the discharge pressure. The small (normal) leakage flow from the discharge end to the suction end due to the clearance between the wear rings and the impeller is described by

$$Q_l = c_l\sqrt{|p_d - p_s|}sign(p_d - p_s), \tag{35}$$

where c_l is a flow coefficient. The discharge flow, Q, is then

$$Q = Q_i - Q_l.$$

Pump temperatures are often monitored as indicators of pump condition. The oil heats up due to the radial and thrust bearings and cools to the environment:

$$\dot{T}_o = \frac{1}{J_o}(H_{o,1}(T_t - T_o) + H_{o,2}(T_r - T_o) - H_{o,3}(T_o - T_a)), \tag{36}$$

where J_o is the thermal inertia of the oil, and the $H_{o,i}$ terms are heat transfer coefficients. The thrust bearings heat up due to the friction between the pump shaft and the bearings, and cool to the oil and the environment:

$$\dot{T}_t = \frac{1}{J_t}(r_t\omega^2 - H_{t,1}(T_t - T_o) - H_{t,2}(T_t - T_a)), \tag{37}$$

where J_t is the thermal inertia of the thrust bearings, r_t is the friction coefficient for the thrust bearings, and the $H_{t,i}$ terms are heat transfer coefficients. The radial bearings behave similarly:

$$\dot{T}_r = \frac{1}{J_r}(r_r\omega^2 - H_{r,1}(T_r - T_o) - H_{r,2}(T_r - T_a)) \tag{38}$$

where J_r is the thermal inertia of the radial bearings, r_r is the friction coefficient for the radial bearings, and the $H_{r,i}$ terms are heat transfer coefficients. Note that r_t and r_r contribute to the overall friction coefficient r.

The overall input vector $\mathbf{u}$ is given by

$$\mathbf{u}(t) = \begin{bmatrix} p_s(t) & p_d(t) & T_a(t) & V(t) & \omega_s(t)\end{bmatrix}^T. \tag{39}$$

The measurement vector $\mathbf{y}$ is given by

$$\mathbf{y}(t) = \begin{bmatrix} \omega(t) & Q(t) & T_t(t) & T_r(t) & T_o(t)\end{bmatrix}^T. \tag{40}$$

The performance constraints of the pump are specified by efficiency and temperature limits:

$$C_1 : \eta(t) > \eta^- \tag{41}$$
$$C_2 : T_o(t) < T_o^+ \tag{42}$$
$$C_3 : T_t(t) < T_t^+ \tag{43}$$
$$C_4 : T_r(t) < T_r^+, \tag{44}$$

where the $-$ subscript denotes a minimum and the $+$ superscript denotes a maximum, and efficiency η is defined as $\eta = (VI)/((p_d - p_s)Q)$. We take $\eta^- = 0.75\eta_0$, where η_0 is the nominal efficiency. When the maximum temperatures are reached, irreversible damage occurs. Here, we use $T_o^+ = 333$ K and $T_t^+ = T_r^+ = 370$ K.

The most significant damage mechanism for pumps is impeller wear, represented as a decrease in impeller area A [20, 21]. Since the impeller area is proportional to b_0, a decrease causes a decrease in the pump pressure, and, hence, the pump efficiency. We use the erosive wear equation [22] to describe how the impeller area changes over time [10]:

$$\dot{A}(t) = -w_A Q_i(t)^2,$$

Table 1. Estimation and Prediction Performance for UKF

$\mathbf{n}$	PRMSE$_{w_A}$	PRMSE$_{w_t}$	PRMSE$_{w_r}$	$\overline{\text{RSD}}_{w_A}$	$\overline{\text{RSD}}_{w_t}$	$\overline{\text{RSD}}_{w_r}$	$\overline{\text{RA}}$	$\overline{\text{RSD}}_{RUL}$
1	2.44	1.95	3.29	10.97	10.05	9.40	97.42	8.39
10	3.36	2.77	3.74	10.56	9.77	9.54	96.67	8.46
100	4.26	2.56	4.48	11.27	9.68	9.77	95.65	9.15
1000	3.82	3.41	4.17	11.91	10.26	11.41	94.84	10.47

Table 2. Estimation and Prediction Performance for PF

$\mathbf{n}$	PRMSE$_{w_A}$	PRMSE$_{w_t}$	PRMSE$_{w_r}$	$\overline{\text{RSD}}_{w_A}$	$\overline{\text{RSD}}_{w_t}$	$\overline{\text{RSD}}_{w_r}$	$\overline{\text{RA}}$	$\overline{\text{RSD}}_{RUL}$
1	4.32	5.73	3.65	12.73	11.77	11.27	96.95	10.65
10	4.99	3.05	3.43	12.63	11.71	11.02	96.38	10.68
100	11.77	4.33	7.27	12.52	12.22	10.90	92.47	10.71
1000	20.54	10.25	19.35	12.43	11.30	11.43	71.52	13.29

where w_A is a wear coefficient. Because A is proportional to b_0, then $\dot{b}_0(t) = k\dot{A}(t) = -kw_A Q_i(t)^2$, so we estimate $b_0(t)$ and $w_{b_0} = kw_A$.

Another significant damage mechanism for pumps is bearing wear, captured as an increase in the friction coefficient. Sliding and rolling friction generate wear of material which increases the coefficient of friction [10, 22]:

$$\dot{r}_t(t) = w_t r_t \omega^2 \tag{45}$$

$$\dot{r}_r(t) = w_r r_r \omega^2, \tag{46}$$

where w_t and w_r are the wear coefficients.

Results

We performed a number of simulation experiments in which the values of wear parameters for the different damage processes were selected randomly in $[1 \times 10^{-3}, 4 \times 10^{-3}]$ at increments of 1×10^{-3} for w_{b_0}, in $[1 \times 10^{-11}, 7 \times 10^{-11}]$ at increments of 0.5×10^{-11} for w_t and w_r, such that the maximum wear rates corresponded to a minimum EOL of 20 hours. The filters had to jointly estimate the state and these unknown parameters. Both algorithms used the variance control algorithm, controlling the RSD of the hidden wear parameter estimates, with $S_j = 2$ and $\mathbf{v}_j^* = [50, 10]$, $\mathbf{T}_j = [60, 0]$, and $\mathbf{P}_j = [1 \times 10^{-3}, 1 \times 10^{-4}]$ for all j. In order to investigate how the algorithms performed under different noise levels, we varied the sensor noise variance by factors of 1, 10, 100, and 1000, and performed 15 experiments for each case, averaging results over each case. The future input of the pump was considered to be known, and it was always operated at a constant RPM. Hence, the only uncertainty present is that involved in the noise terms and that introduced by the filtering algorithms.

The estimation performance is calculated using percent root mean square error (PRMSE) for accuracy of the wear parameter estimates, and RSD for spread. Note that the variance control algorithm attempts to control the RSD of each wear parameter estimate to 10%, so ideally the computed RSD would be exactly that. The RUL prediction performance is calculated using relative accuracy (RA) [11] for accuracy (a score of 100% is best) and RSD for prediction spread. The prediction metrics are computed at each prediction point and averaged over an entire experiment.

The results for the UKF are summarized in Table 1. The wear parameters were estimated with very good accuracy, and with spread around the desired 10%. As sensor noise increased (denoted by the column labeled $\mathbf{n}$ in the table), estimation performance generally became worse in both accuracy and precision, but the variance control algorithm was still able to keep RSD near the desired level, even when the sensor noise variance was increased by a factor of 1000. The good estimation performance translated to good prediction performance since there was no uncertainy in the future inputs. Both RA and RSD of the predictions were good and performance decreased with increased noise, as expected.

The results for the PF are summarized in Table 2. The PF used $N = 500$ particles and assumed the sensor noise variance was 10 times the actual value (this is sometimes called a “roughening penalty” and in this case improves convergence and accuracy). Similar trends are observed here, however, the accuracy is decreased and the variance control algorithm has a more difficult time controlling RSD. As sensor noise increases, the PF has a more difficult time converging, and when the sensor noise variance is increased by a factor of 1000, some experiments did not converge, resulting in very poor estimation and prediction and bringing the overall average down. The UKF, on the other hand, easily converged with all noise levels considered. Performance of the PF could possibly be improved by increasing the number of particles and/or changing the parameters of the variance control algorithm.

The performance of the two filters is easily compared visually. Here, we choose the case where $w_{b_0} = 2 \times 10^{-3}$, $w_t = 3 \times 10^{-11}$, and $w_r = 1 \times 10^{-11}$. Estimation results for the UKF are shown in Fig. 3 and results for the PF are shown in Fig. 4. We show the true values, denoted with the * superscript, the mean estimate, and the minimum and maximum values of the samples. The spread of the UKF estimate appears significantly smaller, but this is only because the sigma points represent the distribution with a set of sigma points whose minimum and maximum values are very close to the mean, whereas with the PF, the samples are stochastically selected, so the minimum and maximum values may be far from the mean. In both cases the effects

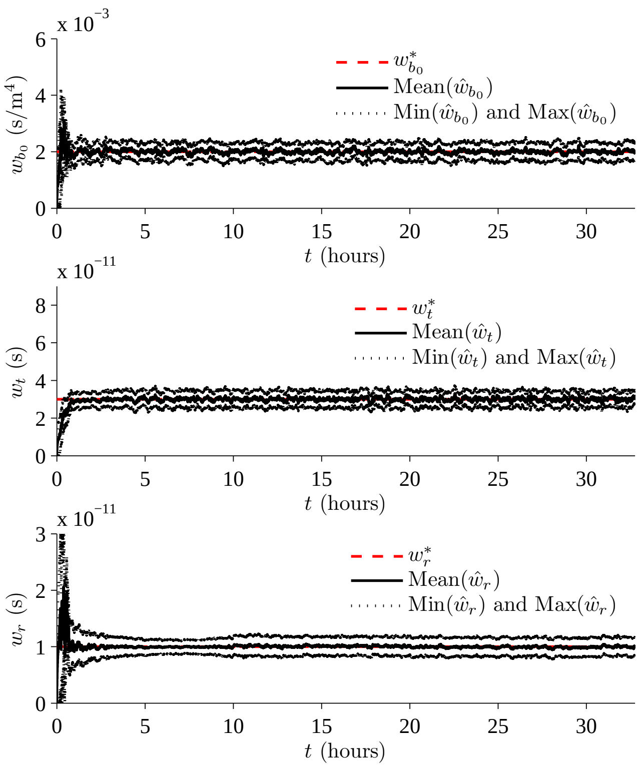

Figure 3. UKF estimation of pump wear parameters.

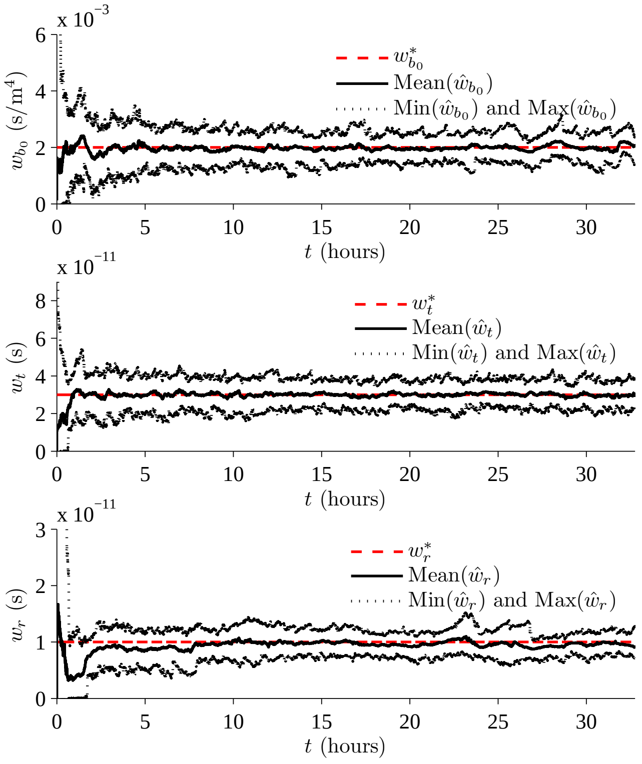

Figure 4. PF estimation of pump wear parameters.

of the variance control algorithm are also observed, e.g., for w_r the initial estimate has a very large variance, but quickly decreases to the desired level of relative spread. Convergence, though, is clearly better with the UKF.

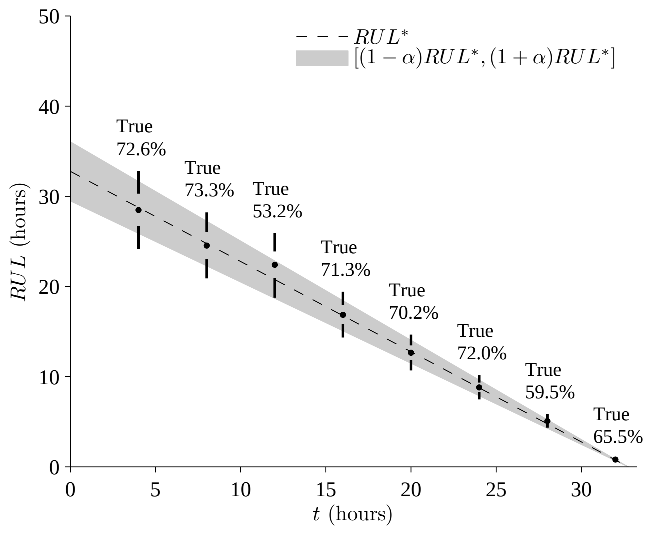

Figure 5. α-λ performance with $\alpha = 0.1$ and $\beta = 0.5$ for the UKF.

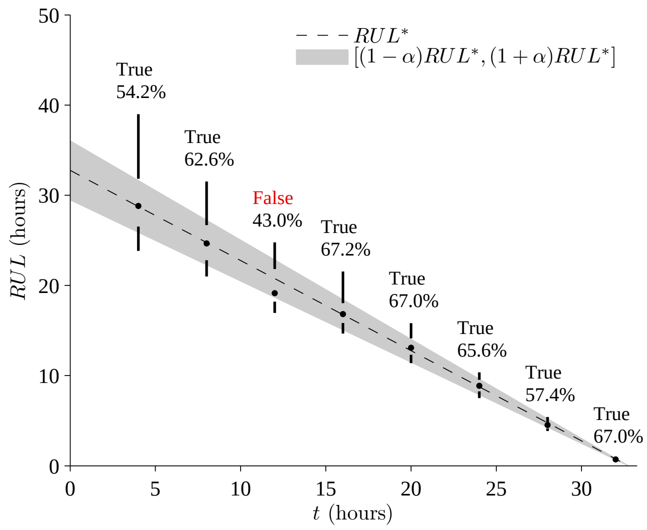

Figure 6. α-λ performance with $\alpha = 0.1$ and $\beta = 0.5$ for the PF.

The prediction results for these cases are shown in Fig. 5 for the UKF and Fig. 6 for the PF using an α-λ plot [11]. With the α-λ metric, we check at each prediction point whether β of the distribution lies within α of the true RUL. In the figure, the accuracy bound defined by $\alpha = 10\%$ is shown as a gray cone, and we select $\beta = 50\%$. For the UKF, the metric is satisfied at all prediction points, with about 70% of the distribution falling within the accuracy bound, with the mean always falling within the bound. For the PF, the mean always falls within the bound, but the portion of the distribution falling within the bound is consistently less than with the UKF, and the metric fails at the third prediction point.

Overall, the UKF consistently performs better than the PF for this case study. Further, it does this at a much reduced computational cost. The complete state-parameter vector has 11 variables, so the UKF with the symmetric transform requires $2 \times 11 + 1 = 23$ sigma points, whereas the PF used 500 particles. This also translates to computational savings in the prediction algorithm, because (*i*) only 23 simulations have to be performed compared to 500, and (*ii*) the minimum values are larger for the UKF than for the PF, and the time to EOL (and thus the time to simulate to EOL) is inversely proportional to the wear rate value. So, prediction using the sigma points is more efficient than using the particles. Note, however, that the unscented transform may be applied to the particle distribution and the resulting sigma points can be simulated forward instead as shown in [15].

Both filters require some degree of tuning. The UKF is generally easier to tune because it has fewer free parameters. The most difficult aspect can be selecting the parameters of the unscented transform. In this case, we used $\kappa = 3 - n_x = 3 - 11 = -8$, as recommended in [6], which worked quite well. With the PF, the number of particles and the roughening penalties must be tuned, but some general heuristics are available [7].

It is expected that the Daum filter, when applicable, can improve performance over approximate filters like the UKF and PF. However, its applicability for prognostics is apparently limited, because many damage progression processes are nonlinear and may not meet the rather strict requirements on the model dynamics. This problem becomes more likely when joint state-parameter estimation must be performed, as introducing unknown parameters as states increases the amount and types of nonlinearities. For example, in the damage progression equation 45 for $\dot{r}_t$, we have effectively three states when performing joint state-parameter estimation: r_t, w_t, and ω. When taking $\frac{\partial r_t}{\partial r_t}$ we end up with $w_t\omega^2$ which would violate equation 4. An approximate version of the Daum filter is available, but still requires solving analytically the PDE of equation 3. For the pump, a solution to this PDE could not be found by Mathematica, as it is not able to solve PDEs analytically for multi-dimensional systems. It is questionable as to whether it is worth the effort to apply the Daum filter when simpler approximate filters, like the UKF, may be used, and whether the possible improvement in accuracy over the UKF is significant enough to justify the additional filter complexity.

6. Conclusions

In this paper, we reviewed nonlinear filtering approaches with application to prognostics, including the Daum filter, the unscented Kalman filter, and the particle filter. In model-based prognostics, joint state-parameter estimation determines the current health state of the system, and this is followed by a prediction algorithm that uses the estimated state as an input. We adopted a centrifugal pump as a case study, and in simulation demonstrated the application of the UKF and PF to the pump. The Daum filter had requirements on the model dynamics that were too strong in order to apply it also to the pump, and seems to have limited applicability to prognostics applications due to the constraints it puts on the model. In this case study, we found the UKF to outperform the PF in damage estimation, and, as a result, the predictions obtained were also superior in both accuracy and precision. The UKF was also easier to tune and had significantly lower computational complexity.

In the future, we would like to further investigate the applicability of the Daum filter and its related exact nonlinear filters to prognostics applications. When applicable, it is important to compare its performance to simpler algorithms like the extended Kalman filter and the UKF relative to the effort in applying the filter. Further, it remains to be seen whether the approximate version of the Daum filter can outperform the UKF, and, if so, if the performance gain is significant enough to justify the increase in development effort of the filter.

Acknowledgments

The funding for this work was provided by the NASA System-wide Safety and Assurance Technologies (SSAT) project.

References

[1] J. R. Celaya, C. Kulkarni, G. Biswas, S. Saha, and K. Goebel, "A model-based prognostics methodology for electrolytic capacitors based on electrical overstress accelerated aging," in *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2011*, Sept. 2011.

[2] D. Chelidze, "Multimode damage tracking and failure prognosis in electromechanical system," in *Proceedings of the SPIE Conference*, vol. 4733, 2002, pp. 1–12.

[3] V. E. Beneš, "Exact finite-dimensional filters for certain diffusions with nonlinear drift," *Stochastics An International Journal of Probability and Stochastic Processes*, vol. 5, no. 1, pp. 65–92, 1981.

[4] F. Daum, "Exact finite-dimensional nonlinear filters," *IEEE Transactions on Automatic Control*, vol. 31, no. 7, pp. 616–622, 1986.

[5] J. A. DeCastro, L. Tang, K. A. Loparo, K. Goebel, and G. Vachtsevanos, "Exact nonlinear filtering and prediction in process model-based prognostics," in *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2009*, Sept. 2009.

[6] S. J. Julier and J. K. Uhlmann, "A new extension of the Kalman filter to nonlinear systems," in *Proceedings of the 11th International Symposium on Aerospace/Defense Sensing, Simulation and Controls*, 1997, pp. 182–193.

[7] M. S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp, "A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking," *IEEE Transactions on Signal Processing*, vol. 50, no. 2, pp. 174–188, 2002.

[8] M. E. Orchard, "A particle filtering-based framework for on-line fault diagnosis and failure prognosis," Ph.D. dissertation, Georgia Institute of Technology, 2007.

[9] B. Saha and K. Goebel, "Modeling Li-ion battery capacity depletion in a particle filtering framework," in *Proceedings of the Annual Conference of the Prognostics and Health Management Society 2009*, Sept. 2009.

[10] M. Daigle and K. Goebel, "Multiple damage progression paths in model-based prognostics," in *Proceedings of the 2011 IEEE Aerospace Conference*, Mar. 2011.

[11] A. Saxena, J. Celaya, B. Saha, S. Saha, and K. Goebel, "Metrics for offline evaluation of prognostic performance," *International Journal of Prognostics and Health Management (IJPHM)*, vol. 1, 2010.

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

[13] G. Kitagawa, "Monte Carlo filter and smoother for non-Gaussian nonlinear state space models," *Journal of*

Computational and Graphical Statistics, vol. 5, no. 1, pp. 1–25, 1996.

[14] J. Liu and M. West, "Combined parameter and state estimation in simulation-based filtering," *Sequential Monte Carlo Methods in Practice*, pp. 197–223, 2001.

[15] M. Daigle and K. Goebel, "Improving computational efficiency of prediction in model-based prognostics using the unscented transform," in *Annual Conference of the Prognostics and Health Management Society 2010*, Oct. 2010.

[16] ——, "Model-based prognostics under limited sensing," in *2010 IEEE Aerospace Conference*, Mar. 2010.

[17] S. E. Lyshevski, *Electromechanical Systems, Electric Machines, and Applied Mechatronics*. CRC, 1999.

[18] A. Wolfram, D. Fussel, T. Brune, and R. Isermann, "Component-based multi-model approach for fault detection and diagnosis of a centrifugal pump," in *Proceedings of the 2001 American Control Conference*, vol. 6, 2001, pp. 4443–4448.

[19] C. Kallesøe, "Fault detection and isolation in centrifugal pumps," Ph.D. dissertation, Aalborg University, 2005.

[20] G. Biswas and S. Mahadevan, "A hierarchical model-based approach to systems health management," in *2007 IEEE Aerospace Conference*, Mar. 2007.

[21] F. Tu, S. Ghoshal, J. Luo, G. Biswas, S. Mahadevan, L. Jaw, and K. Navarra, "PHM integration with maintenance and inventory management systems," in *Proc. of the 2007 IEEE Aerospace Conference*, Mar. 2007.

[22] I. M. Hutchings, *Tribology: friction and wear of engineering materials*. CRC Press, 1992.

BIOGRAPHY

Matthew Daigle received the B.S. degree in Computer Science and Computer and Systems Engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2004, and the M.S. and Ph.D. degrees in Computer Science from Vanderbilt University, Nashville, TN, in 2006 and 2008, respectively. From September 2004 to May 2008, he was a Graduate Research Assistant with the Institute for Software Integrated Systems and Department of Electrical Engineering and Computer Science, Vanderbilt University, Nashville, TN. During the summers of 2006 and 2007, he was an intern with Mission Critical Technologies, Inc., at NASA Ames Research Center. From June 2008 to December 2011, he was an Associate Scientist with the University of California, Santa Cruz, at NASA Ames Research Center. Since January 2012, he has been with NASA Ames Research Center as a Research Computer Scientist. His current research interests include physics-based modeling, model-based diagnosis and prognosis, simulation, and hybrid systems.

Bhaskar Saha received the Ph.D. and M.S. degrees in electrical and computer engineering from Georgia Institute of Technology, and earned his bachelor's degree in electrical engineering from the Indian Institute of Technology in Kharagpur, India. Dr. Saha is with the Palo Alto Research Center (PARC) as a Member of the Research Staff, with a focus on applying Bayesian inference techniques expertise to classification, state estimation, and prediction problems for intelligent system design and health management. Before joining PARC, he was with Mission Critical Technologies, Inc., at NASA Ames Research Center as a research scientist at the Prognostics Center of Excellence. While there, he developed an integrated Bayesian framework to determine remaining-useful-life probability densities. He also built a hardware-in-the-loop testbed to benchmark his prognostic algorithm for a battery health management system for electric UAVs, which successfully provided real-time battery status and remaining useful life during multiple flight tests. In addition to organizing conferences and sessions for several IEEE events and the Prognostics and Health Management Society, Dr. Saha has won three best-paper awards.

Kai Goebel received the degree of Diplom-Ingenieur from the Technische Universität München, Germany in 1990. He received the M.S. and Ph.D. from the University of California at Berkeley in 1993 and 1996, respectively. Dr. Goebel is currently a deputy branch chief of the Discovery and Systems Health Technology Area at NASA Ames Research Center. He also coordinates the Prognostics Center of Excellence and is the Technical Lead for Prognostics and Decision Making in NASAs System-wide Safety and Assurance Technologies Project. Prior to joining NASA in 2006, he was a senior research scientist at General Electric Corporate Research and Development Center since 1997. He has carried out applied research in the areas of real time monitoring, diagnostics, and prognostics and he has fielded numerous applications for aircraft engines, transportation systems, medical systems, and manufacturing systems. He holds 15 patents and has co-authored more than 200 technical papers in the field of Systems Health Management. Dr. Goebel was an adjunct professor of the CS Department at Rensselaer Polytechnic Institute (RPI), Troy, NY, between 1998 and 2005 where he taught classes in Soft Computing and Applied Intelligent Reasoning Systems. He is currently member of the board of directors of the PHM Society and associate editor of the International Journal of PHM.