Distributed Damage Estimation for Prognostics based on Structural Model Decomposition

Matthew Daigle [1] Anibal Bregon [2] and Indranil Roychoudhury [3]

[1] *University of California, Santa Cruz, NASA Ames Research Center, Moffett Field, CA, 94035, USA*
matthew.j.daigle@nasa.gov
[2] *Department of Computer Science, University of Valladolid, Valladolid, 47011, Spain*
anibal@infor.uva.es
[3] *SGT, Inc., NASA Ames Research Center, Moffett Field, CA, 94035, USA*
indranil.roychoudhury@nasa.gov

ABSTRACT

Model-based prognostics approaches capture system knowledge in the form of physics-based models of components that include how they fail. These methods consist of a damage estimation phase, in which the health state of a component is estimated, and a prediction phase, in which the health state is projected forward in time to determine end of life. However, the damage estimation problem is often multi-dimensional and computationally intensive. We propose a model decomposition approach adapted from the diagnosis community, called possible conflicts, in order to both improve the computational efficiency of damage estimation, and formulate a damage estimation approach that is inherently distributed. Local state estimates are combined into a global state estimate from which prediction is performed. Using a centrifugal pump as a case study, we perform a number of simulation-based experiments to demonstrate the approach.

1. INTRODUCTION

Model-based prognostics approaches capture knowledge of how a system and its components fail through the use of physics-based models that capture the underlying physical phenomena (Daigle & Goebel, 2010b; Saha & Goebel, 2009; Luo, Pattipati, Qiao, & Chigusa, 2008). Model-based prognostics algorithms consist of two parts: (*i*) *damage estimation*, which is fundamentally a joint state-parameter estimation problem, and (*ii*) *prediction*, which projects the current joint state-parameter estimate forward in time to determine end of life (EOL). In (Daigle & Goebel, 2011), we developed a prognostics framework using particle filters for the damage estimation step that handles several simultaneously progressing damage processes. However, the approach may not scale well as the number of damage processes to track (i.e., the dimension of the system) increases.

In this paper, we improve both the scalability and the computational efficiency of the damage estimation task by exploiting structural model decomposition (Williams & Millar, 1998), similar to methods developed within the diagnosis community (Bregon, Pulido, & Biswas, 2009; Roychoudhury, Biswas, & Koutsoukos, 2009; Staroswiecki & Declerck, 1989). In particular, we adopt the possible conflicts (PCs) approach (Pulido & Alonso-González, 2004). PCs decompose a global system model into minimal overdetermined subsystems (local submodels) for fault detection and isolation (Pulido & Alonso-González, 2004). PCs have also been used to formulate smaller estimation tasks for fault identification (Bregon, Pulido, & Biswas, 2009). In general, PCs can be used to automatically decompose a global joint state-parameter estimation task into a set of local estimation tasks that are easier to solve and require less overall computation. We use the PC approach to derive a minimal set of submodels and define a local damage estimation task for each one. Every local estimator computes a local joint state-parameter estimate, represented as a probability distribution. Then, the local estimates are merged into a global estimate from which prediction is performed in the typical way (Daigle & Goebel, 2011).

The models are decomposed into independent submodels by using measured signals as local inputs. Therefore, each local estimator operates independently, and the damage estimation becomes naturally distributed. Clearly then, this approach establishes a formal basis for distributed prognostics. This is in contrast to other proposed distributed prognostics approaches, e.g. (Saha, Saha, & Goebel, 2009), which still treat the prognostics problem as a global one in which only the computation is distributed, whereas we propose to decompose the global problem into a set of local ones for which computation may be trivially distributed.

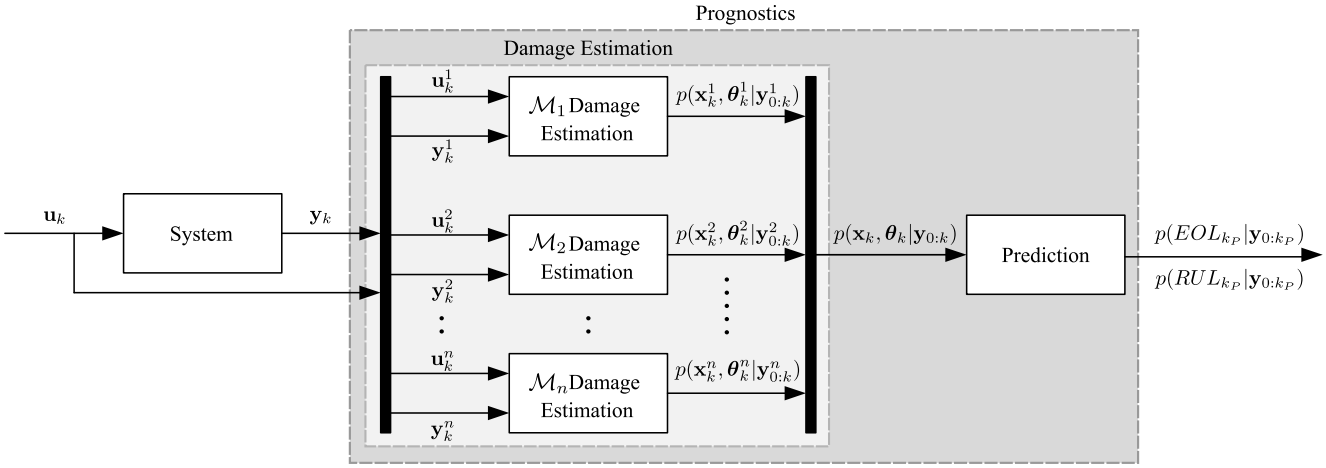


Figure 1. Prognostics architecture.

We demonstrate our prognostics methodology on a centrifugal pump. Centrifugal pumps appear in a variety of domains and often need to be operated for long time periods, hence diagnostics and prognostics become critical to ensuring continued operation that meets performance requirements. We apply our model-based prognostic approach based on structural model decomposition to centrifugal pumps using a number of simulation-based experiments when multiple damage mechanisms are active, and compare to results using the global estimation approach presented in (Daigle & Goebel, 2011).

The paper is organized as follows. Section 2 formally defines the prognostics problem and describes the prognostics architecture. Section 3 describes the modeling methodology and develops the centrifugal pump model for prognostics. Section 4 presents the model decomposition approach and provides results for the pump model. Section 5 describes the particle filter-based local damage estimation method. Section 6 discusses the prediction methodology. Section 7 provides results from simulation-based experiments and evaluates the approach. Section 8 concludes the paper.

2. Prognostics Approach

The goal of prognostics is the prediction of EOL and/or remaining useful life (RUL) of a component. In this section, we first formally define the problem of prognostics. We then describe the model-based prognostics architecture based on structural model decomposition.

2.1 Problem Formulation

In general, we define a system model 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 $t \in \mathbb{R}$ is a continuous time variable, $\mathbf{x}(t) \in \mathbb{R}^{n_x}$ is the state vector, $\boldsymbol{\theta}(t) \in \mathbb{R}^{n_\theta}$ is the 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. The parameters $\boldsymbol{\theta}(t)$ evolve in an unknown way.

The goal is to predict EOL (and/or RUL) at a given time point t_P using the discrete sequence of observations up to time t_P, denoted as $\mathbf{y}_{0:t_P}$. The component must meet a given set of functional requirements. We say the component has failed when it no longer meets one of these requirements. In general, we may capture this boundary on acceptable component behavior using a threshold that is a function of the system state and parameters, $T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t))$, where $T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 1$ if the system has failed and $T_{EOL}(\mathbf{x}(t), \boldsymbol{\theta}(t)) = 0$ otherwise. 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 the damage threshold is met. RUL is then

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

Due to the noise terms $\mathbf{v}(t)$ and $\mathbf{n}(t)$, and uncertainty in the future inputs of the system, we at best compute only a probability distribution of the EOL or RUL, i.e., $p(EOL(t_P)|\mathbf{y}_{0:t_P})$ or $p(RUL(t_P)|\mathbf{y}_{0:t_P})$.

2.2 Prognostics Architecture

In our model-based approach, we develop detailed physics-based models of components and systems that include descriptions of how faults and damage evolves in time. These models depend on unknown parameters $\boldsymbol{\theta}(t)$. Therefore, damage estimation is fundamentally a joint state-parameter estimation problem. In discrete time k, we jointly estimate $\mathbf{x}_k$ and $\boldsymbol{\theta}_k$, and use these estimates to predict EOL and RUL at desired time points. Here, we assume that prognostics is

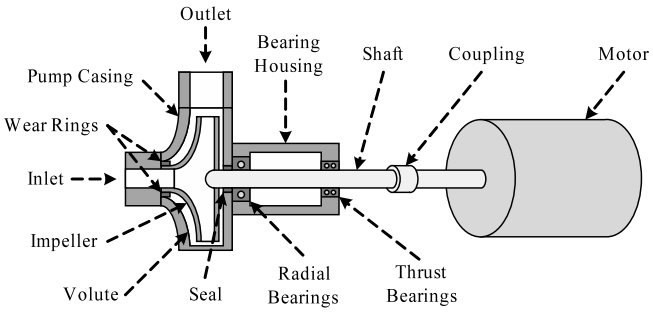


Figure 2. Centrifugal pump.

not aided by a fault diagnosis module, and so we must jointly estimate all possible damage modes.

We employ the prognostics architecture in Fig. 1. The system is provided with inputs $\mathbf{u}_k$ and provides measured outputs $\mathbf{y}_k$. The damage estimation module takes as input both $\mathbf{u}_k$ and $\mathbf{y}_k$, and produces the estimate $p(\mathbf{x}_k, \boldsymbol{\theta}_k|\mathbf{y}_{0:k})$. In this work, we decompose the global damage estimation problem into several subproblems based on model decomposition, as shown in Fig. 1. A model decomposition algorithm splits the global model into n submodels, $\mathcal{M}_1, \mathcal{M}_2, \ldots, \mathcal{M}_n$. We construct for each submodel $\mathcal{M}_i$ a local estimator that performs damage estimation. Each estimator has input $\mathbf{u}_k^i \subseteq \mathbf{u}_k$ and $\mathbf{y}_k^i \subseteq \mathbf{y}_k$ and produces the local state estimate $p(\mathbf{x}_k^i, \boldsymbol{\theta}_k^i|\mathbf{y}_{0:k}^i)$, where $\mathbf{x}_k^i \subseteq \mathbf{x}_k$ and $\boldsymbol{\theta}_k^i \subseteq \boldsymbol{\theta}_k$. Note that for two submodels $\mathcal{M}_i$ and $\mathcal{M}_j$, in general it is possible that $\mathbf{x}_k^i \cap \mathbf{x}_k^j \neq \varnothing$ and $\boldsymbol{\theta}_k^i \cap \boldsymbol{\theta}_k^j \neq \varnothing$. The local estimates are merged into the global estimate $p(\mathbf{x}_k, \boldsymbol{\theta}_k|\mathbf{y}_{0:k})$. The prediction module uses this joint state-parameter distribution, 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.

3. PUMP MODELING

We apply our prognostics approach to a centrifugal pump, and develop a physics-based model of its nominal and faulty behavior. Centrifugal pumps are used in a variety of domains for fluid delivery. A schematic of a typical centrifugal pump is shown in Fig. 2. Fluid enters the inlet, and the rotation of the impeller, driven by an electric motor, forces fluid through the outlet. Radial and thrust bearings, along with lubricating oil contained within the bearing housing, helps to minimize friction along the pump shaft. Wear rings prevent internal pump leakage from the outlet to the inlet side of the impeller, but a small clearance is typically allowed to minimize friction (a small internal leakage is normal). We review here the main features of the model, and refer the reader to (Daigle & Goebel, 2011) for details.

The state of the pump is given by

$$\mathbf{x}(t) = \begin{bmatrix} \omega(t) & T_t(t) & T_r(t) & T_o(t) \end{bmatrix}^T,$$

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 torque balance,

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

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.

We assume the pump is driven by an induction motor with a polyphase supply. A torque is produced on the rotor only when there is a difference, i.e., a *slip*, between the synchronous speed of the supply voltage, ω_s and the mechanical rotation, ω. Slip, s, is defined as

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

The expression for the torque τ_e is derived from an equivalent circuit representation for the three-phase induction motor, based on rotor and stator resistances and inductances and the slip s (Lyshevski, 1999):

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

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. The dependence of torque on slip creates a feedback loop that causes the rotor to follow the rotation of the magnetic field. The rotor speed may be controlled by changing the input frequency ω_s.

The load torque τ_L is a polynomial function of the flow rate through the pump and the impeller rotational velocity (Kallesøe, 2005):

$$\tau_L = a_0\omega^2 + a_1\omega Q - a_2 Q^2,$$

where Q is the flow, and a_0, a_1, and a_2 are coefficients derived from the pump geometry (Kallesøe, 2005).

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

$$p_p = A\omega^2 + b_1\omega Q - b_2 Q^2,$$

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:

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

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

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

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

$$Q = Q_i - Q_l.$$

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

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

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:

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

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:

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

where the parameters here take on analogous definitions.

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

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

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

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

Fig. 3 shows nominal pump operation. The input voltage (and frequency) are varied to control the pump speed. The electromagnetic torque is produced initially as slip is 1. This causes a rotation of the motor to match the rotation of the magnetic field, with a small amount of slip remaining, depending on how large the load and friction torques are. As the pump rotates, fluid flow is created. The bearings heat up as the pump rotates and cool when the pump rotation slows.

3.1 Damage Modeling

For the purposes of prognostics, the model must include *damage variables* $\mathbf{d} \subseteq \mathbf{x}$ representing the amount of particular forms of damage. The most significant forms of damage for pumps are impeller wear, caused by cavitation and erosion by the flow, and bearing failure, caused by friction-induced wear of the bearings. In each case, we map the damage to a particular parameter in the nominal model, and this parameter becomes a state variable in $\mathbf{d}(t)$. The evolution of these damage variables is described by *damage progression equations*,

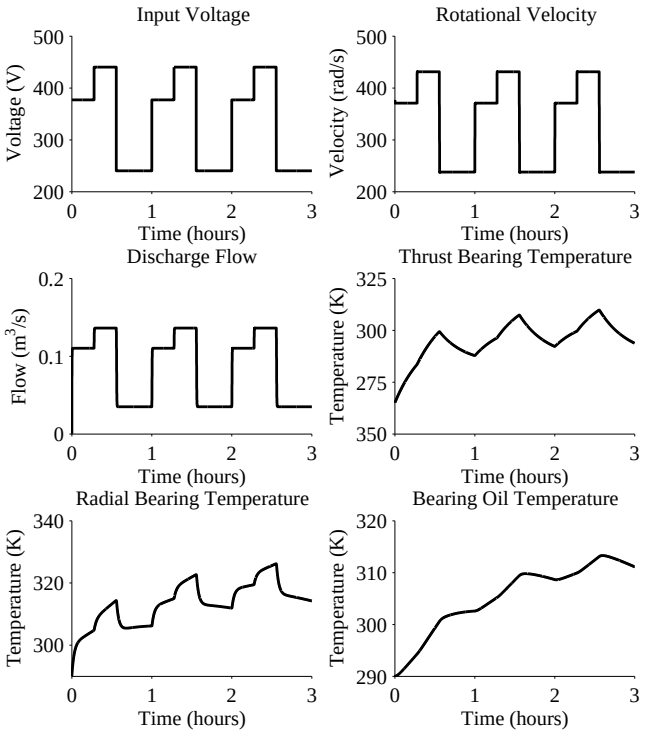


Figure 3. Nominal pump operation.

which are included in the state equation $\mathbf{f}$. These equations are parameterized by unknown *wear parameters* $\mathbf{w} \subseteq \boldsymbol{\theta}$.

Impeller wear is represented as a decrease in effective impeller area A (Biswas & Mahadevan, 2007; Daigle & Goebel, 2011). We use the erosive wear equation (Hutchings, 1992):

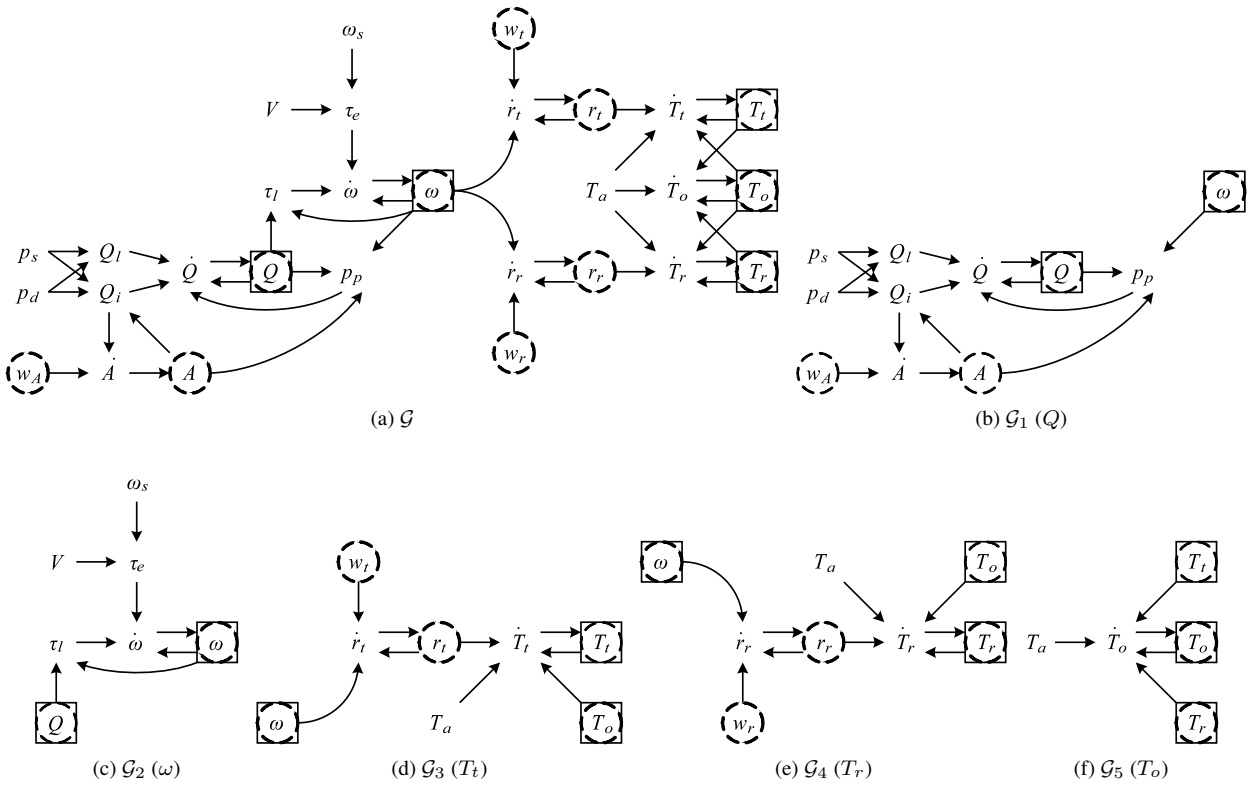
$$\dot{A}(t) = -w_A Q_i^2,$$

where w_A is a wear coefficient. A decrease in the impeller area will decrease the pump pressure, which, in turn, reduces the delivered flow, and, therefore, pump efficiency. The pump must operate at a certain minimal efficiency, defining an EOL criteria. We define A^- as the minimum value of the impeller area at which this requirement is met, hence, $T_{EOL} = 1$ if $A(t) < A^-$.

Bearing wear is captured as an increase in friction. Sliding and rolling friction generate wear of material which increases the effective coefficient of friction (Hutchings, 1992; Daigle & Goebel, 2010b, 2011):

$$\dot{r}_t(t) = w_t r_t \omega^2,$$
$$\dot{r}_r(t) = w_r r_r \omega^2,$$

where w_t and w_r are the wear coefficients. The slip compensation provided by the electromagnetic torque generation will mask small changes in friction, but these changes can be observed using the bearing temperatures. Limits on the maximum values of these temperatures define EOL for bearing wear. We define r_t^+ and r_r^+ as the maximum permissible values of the friction coefficients, before the temperature limits



(a) $\mathcal{G}$ (b) $\mathcal{G}_1$ (Q) (c) $\mathcal{G}_2$ (ω) (d) $\mathcal{G}_3$ (T_t) (e) $\mathcal{G}_4$ (T_r) (f) $\mathcal{G}_5$ (T_o)

Figure 4. System graph and minimal subgraphs of the pump.

are exceeded over a typical usage cycle. So, $T_{EOL} = 1$ if $r_t(t) > r_t^+$ or $r_r(t) > r_r^+$.

So, the damage variables are given by

$$\mathbf{d}(t) = \begin{bmatrix} A(t) & r_t(t) & r_r(t) \end{bmatrix}^T,$$

and the full state vector becomes

$$\mathbf{x}(t) = \begin{bmatrix} \omega(t) & T_t(t) & T_r(t) & T_o(t) & A(t) & r_t(t) & r_r(t) \end{bmatrix}^T.$$

The wear parameters form the unknown parameter vector, i.e.,

$$\mathbf{w}(t) = \boldsymbol{\theta}(t) = \begin{bmatrix} w_A & w_t & w_r \end{bmatrix}^T.$$

4. Model Decomposition

Especially for nonlinear systems, state and parameter estimation is a nontrivial problem for which no general closed-form solution exists. In general, these problems may be solved by numerical optimization methods, where the computational complexity is exponential in the size of the model, or by approximate Bayesian filtering methods, like particle filters, where the computational complexity is only linear in the size of the sample space, but, the number of sufficient samples grows with the model size.

Several approaches have been developed to decrease the computational complexity by model decomposition, in which the global model is decomposed into several independent submodels (Staroswiecki & Declerck, 1989; Williams & Millar, 1998). This results in a set of smaller, lower-dimensional estimation tasks. We adopt the PC approach for model decomposition. PCs are minimal subsets of equations with sufficient analytical redundancy to generate fault hypotheses from observed measurement deviations, and, in previous work, we used PCs to propose a more robust and computationally simpler parameter estimation approach for fault identification (Bregon, Pulido, & Biswas, 2009), where the parameter estimation task using the entire system model was replaced by a set of smaller estimation problems (one for each PC). In this approach, fault identification is fundamentally a joint state-parameter estimation problem. This is equivalent to the damage estimation problem in prognostics, only the models specifically include damage progression. So, in this paper, we adopt this paradigm for distributed damage estimation.

In order to compute the minimal set of submodels of a system, a structural representation of the system model is needed. In previous work, we computed submodels as PCs using hypergraphs (Pulido & Alonso-González, 2004) or Temporal Causal Graphs (TCGs) (Bregon, Pulido, & Biswas, 2009) as inputs. Representations that include computational causality, like TCGs, are favored because causality allows efficient derivation of PCs. In this work, we represent the system model with a directed hypergraph, and use an algorithm close to the TCG-based algorithm presented in (Bregon, Pulido, Biswas, & Koutsoukos, 2009).

The system model $\mathcal{M}$ is represented using a set of functions $\mathcal{F}$ over variables V, where a subset of the variables $X \subseteq V$ corresponds to the state variables $\mathbf{x}$, a subset $\Theta \subseteq V$ corresponds to the unknown parameters $\boldsymbol{\theta}$, a subset $U \subseteq V$ corresponds to the (known) inputs $\mathbf{u}$, and a subset $Y \subseteq V$ corresponds to the (measured) outputs $\mathbf{y}$. For the purposes of the model decomposition algorithm, we represent $\mathcal{M}$ using an extended directed hypergraph $\mathcal{G} = (V, E, F)$, where V is the set of vertices corresponding directly to the variables in $\mathcal{M}$, E is the set of hyperedges of the form (V', v) with $V' \subseteq V$ being a set of vertices and $v \in V$ being a single vertex, and F is a map from an edge $(V', v) \in E$ to a function $f \in \mathcal{F}$ such that $v = f(v_1, v_2, \ldots, v_n)$ where $V' = \{v_1, v_2, \ldots, v_n\}$. $\mathcal{M}$ and $\mathcal{G}$ are equivalent data structures, where the direction of the edges in $\mathcal{G}$ captures the computational causality of the functions in $\mathcal{F}$. The variable sets X, Θ, Y, and U are represented equivalently in $\mathcal{G}$ as vertex sets, and we use $y_i \in Y$ to refer to the vertex/variable corresponding to the ith output in $\mathbf{y}$.

Fig. 4a shows the hypergraph $\mathcal{G}$ for the pump model described in Section 3. Individual arrows pointing to the same vertex are to be interpreted as a single directed hyperedge. State variables are denoted using dashed circles, and measured variables are denoted with boxes.

Algorithm 1 computes subgraphs corresponding to PCs from a system graph $\mathcal{G}$. One PC will be computed for each output $y_i \in Y$. So, for the pump model, there will be five total submodels, one each for ω, Q, T_t, T_r, and T_o. For each vertex in Y, the algorithm propagates back to members of U and Y, which will be used as inputs to the submodel. Vertices which are not included in U or Y or have not yet been included in V_i are added to the set $vertices$ for further backward propagation. For example, starting with T_t in Fig. 4a to form the subgraph shown in Fig. 4d, we propagate to T_o, a measured output, at which propagation terminates, and $\dot{T}_t$. From $\dot{T}_t$ we propagate back further to the input T_a and the state r_t, from which we propagate to $\dot{r}_t$, from which we propagate to measured output ω and the unknown parameter w_t. All vertices and edges encountered are added to V_i and E_i, respectively, and the model equations corresponding to the added edges are included in the submodel as well. The algorithm forms from $\mathcal{G}$ subgraphs $\mathcal{G}_i = (V_i, E_i, F_i)$, which may be easily translated to submodels $\mathcal{M}_i$. If each $v \in X \cup \Theta$ is causally linked to at least one output, then every variable $x \in X$ and $\theta \in \Theta$ will belong to at least one V_i over the set of $\mathcal{G}_i$ computed, i.e., will belong to at least one submodel $\mathcal{M}_i$.

The algorithm decomposes the pump model into 5 submodels, with their corresponding subgraphs shown as Figs. 4b to 4f. For example, the T_t subgraph takes as input measurements of ω and T_o, and computes the expected value of T_t. Damage estimation for this submodel will compute estimates of T_t, r_t, and w_t. Note that, for the pump model, each state or parameter will be estimated by exactly one submodel, therefore, there will be no overlap in the local estimates.

Algorithm 1 $\{\mathcal{G}_i\}_{i=1}^{n_y} = \texttt{Decompose}(\mathcal{G})$

```
for i = 1 to n_y do
  V_i ← {y_i}
  E_i ← ∅
  vertices ← {y_i}
  while vertices ≠ ∅ do
    v ← vertices{1}
    vertices ← vertices \ {v}
    edges ← {(V', v) ∈ E : V' ⊆ V}
    for all (V', v) ∈ edges do
      for all v' ∈ V' do
        if v' ⊈ U and v' ⊈ Y and v' ⊈ V_i then
          vertices ← vertices ∪ {v'}
        end if
      end for
      V_i ← V_i ∪ V'
      E_i ← E_i ∪ {(V', v)}
      F_i(V', v) ← F(V', v)
    end for
  end while
  G_i ← (V_i, E_i, F_i)
end for
```

5. DAMAGE ESTIMATION

In our local estimation scheme, the local estimator for each submodel $\mathcal{M}_i$ produces a local estimate $p(\mathbf{x}_k^i, \boldsymbol{\theta}_k^i | \mathbf{y}_{0:k})$, where $\mathbf{x}_k^i \subseteq \mathbf{x}_k$ and $\boldsymbol{\theta}_k^i \subseteq \boldsymbol{\theta}_k$. The local estimates are combined into the global state estimate $p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k})$.

Due to the decoupling introduced by the decomposition scheme, we lose information about the covariance between states in separate submodels. If these covariances are nominally small, then this information loss is acceptable and the approximation of the global state estimate obtained by merging the local state estimates into a global state estimate will closely approximate the global estimate obtained through a global damage estimator. Although we lose information due to the decoupling, the advantage is that the local estimation tasks are naturally distributed, and therefore, unlike the global estimation approach, the distributed approach scales well as the size of the model increases. Further, the local estimation tasks become easier to solve and should require less computational resources without sacrificing estimation performance. This should be the case as long as the sensor measurements that are used as inputs to the submodels are reliable and do not exhibit extremely high noise.

A general solution to the problem of damage estimation is the *particle filter*, which may be directly applied to nonlinear systems with non-Gaussian noise terms (Arulampalam, Maskell, Gordon, & Clapp, 2002). The main disadvantage of the particle filter is the computational complexity, which is linear in the amount of samples, or *particles*, that are used to approximate the state distribution, as typically a large number of particles are needed, and the sufficient number of particles increases with the dimension of the state-parameter space. A

key advantage of the model decomposition algorithm is that it creates submodels that are simpler than the global model. Some of these submodels may be completely linear, and some may require only state estimation, not joint state-parameter estimation. If only state estimation is required, the Kalman filter or one of its nonlinear extensions may be used, which requires computation on the order of a particle filter using one particle (e.g., Kalman filter or extended Kalman filter) or a number of particles linear in the state dimension (e.g., unscented Kalman filter (Julier & Uhlmann, 1997)), resulting in a significant improvement in computational complexity. The remaining submodels that require joint state-parameter estimation represent several small, low-dimensional estimation problems that are easier to solve than the global one, therefore requiring less computation overall.

In particle filters, the state distribution is approximated by a set of discrete weighted samples, or particles:

$$\{(\mathbf{x}_k^{(i)}, \boldsymbol{\theta}_k^{(i)}), w_k^{(i)}\}_{i=1}^N,$$

where N denotes the number of particles, and for particle i, $\mathbf{x}_k^{(i)}$ denotes the state vector estimate, $\boldsymbol{\theta}_k^{(i)}$ denotes the parameter vector estimate, and $w_k^{(i)}$ denotes the weight. The posterior density is approximated by

$$p(\mathbf{x}_k, \boldsymbol{\theta}_k | \mathbf{y}_{0:k}) \approx \sum_{i=1}^{N} w_k^{(i)} \delta_{(\mathbf{x}_k^{(i)}, \boldsymbol{\theta}_k^{(i)})}(d\mathbf{x}_k d\boldsymbol{\theta}_k),$$

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

We use the sampling importance resampling (SIR) particle filter, using systematic resampling. Each particle is propagated forward to time k by first sampling new parameter values, and then 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. Pseudocode is provided in (Arulampalam et al., 2002; Daigle & Goebel, 2011).

The parameters $\boldsymbol{\theta}_k$ evolve by some unknown random process that is independent of the state $\mathbf{x}_k$. To perform parameter estimation within a particle filter framework, we assign a random walk evolution, i.e., $\boldsymbol{\theta}_k = \boldsymbol{\theta}_{k-1} + \boldsymbol{\xi}_{k-1}$, where $\boldsymbol{\xi}_{k-1}$ is a noise vector. During the sampling step, particles are generated with parameter values that will be different from the current values of the parameters. The particles with parameter values closest to the true values should match the outputs better, and therefore be assigned higher weight. Resampling will cause more particles to be generated with similar values, so the particle filter converges to the true values as the process is repeated over each step of the algorithm. In general, though, convergence is not always guaranteed.

The selected variance of the random walk noise determines both the rate of this convergence and the estimation performance after convergence. Therefore, this parameter should be tuned to obtain the best possible performance, but the optimal value is dependent on the value of the hidden wear parameter, which is unknown. We use the variance control method presented in (Daigle & Goebel, 2011). In this approach, the variance of the hidden wear parameter estimate is controlled to a user-specified range by modifying the random walk noise variance. We assume that the $\boldsymbol{\xi}$ values are tuned initially based on the maximum expected wear rates. The algorithm uses relative median absolute deviation (RMAD) as the measure of spread. The adaptation scheme controls the error between the actual RMAD of a parameter $\boldsymbol{\theta}(j)$, denoted as v_j, and the desired RMAD value (e.g., 10%), denoted as v_j^*, using a proportional control strategy governend by a gain P (e.g., 1×10^{-3}). There are two different setpoints. The first allows for a convergence period, with setpoint v_{j0}^* (e.g., 50%). Once v_j reaches T (e.g., $1.2v_{j0}^*$), a new setpoint $v_{j\infty}^*$ (e.g., 10%) is established. The advantage of this methodology is that the random walk variance is automatically tuned to achieve the best performance for the requested relative spread for the actual value of the hidden parameter.

6. PREDICTION

Prediction is initiated at a given time k_P. In order to obtain a prediction that is valid for the global state-parameter vector, we must first combine the local estimates into a global estimate. To do this, we assume that the local and global state estimates may be sufficiently approximated by a multivariate normal distribution $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$. So, for each local state-parameter distribution i, we obtain the mean $\boldsymbol{\mu}^i$ and covariance matrix $\boldsymbol{\Sigma}^i$. We then combine all of these into a global mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$. If there is overlap in the state-parameter estimates, i.e., if two submodels both estimate the same state variable x or parameter θ, then we take the average value for common means and covariances (alternate strategies may also be used). The covariance information lost due to the decoupling will appear as zeros in the global covariance matrix.

We then sample from the global state-parameter distribution defined by $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ a number of times to obtain a set of samples that sufficiently approximates the distribution defined by those parameters, which may each be simulated to EOL. An alternate approach is to use the unscented transform to deterministically select the minimal number of samples from this distribution that capture the statistical moments as described in (Daigle & Goebel, 2010a). Although the latter method is computationally more efficient, we use the former method here in order to obtain a fair comparison to the results presented for the global estimation approach in (Daigle & Goebel, 2011).

Using the global joint state-parameter estimate at k_P, $p(\mathbf{x}_{k_P}, \boldsymbol{\theta}_{k_P} | \mathbf{y}_{0:k_P})$, which represents the current knowledge of the system at time k_P, the goal is to compute $p(EOL_{k_P} | \mathbf{y}_{0:k_P})$ and $p(RUL_{k_P} | \mathbf{y}_{0:k_P})$. As discussed in

Algorithm 2 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

Section 5, the particle filter computes

$$p(\mathbf{x}_{k_P}, \boldsymbol{\theta}_{k_P}|\mathbf{y}_{0:k_P}) \approx \sum_{i=1}^{N} w_{k_P}^{(i)} \delta_{(\mathbf{x}_{k_P}^{(i)}, \boldsymbol{\theta}_{k_P}^{(i)})}(d\mathbf{x}_{k_P} d\boldsymbol{\theta}_{k_P}).$$

We can approximate a prediction distribution n steps forward as (Doucet, Godsill, & Andrieu, 2000)

$$p(\mathbf{x}_{k_P+n}, \boldsymbol{\theta}_{k_P+n}|\mathbf{y}_{0:k_P}) \approx \sum_{i=1}^{N} w_{k_P}^{(i)} \delta_{(\mathbf{x}_{k_P+n}^{(i)}, \boldsymbol{\theta}_{k_P+n}^{(i)})}(d\mathbf{x}_{k_P+n} d\boldsymbol{\theta}_{k_P+n}).$$

So, for a particle i propagated n steps forward without new data, we may take its weight as $w_{k_P}^{(i)}$. Similarly, we can approximate the EOL as

$$p(EOL_{k_P}|\mathbf{y}_{0:k_P}) \approx \sum_{i=1}^{N} w_{k_P}^{(i)} \delta_{EOL_{k_P}^{(i)}}(dEOL_{k_P}).$$

To compute EOL, then, we simulate each particle forward to its own EOL and use that particle's weight at k_P for the weight of its EOL prediction. The pseudocode for the prediction procedure is given as Algorithm 2 (Daigle & Goebel, 2010b). Each particle i is propagated forward until $T_{EOL}(\mathbf{x}_k^{(i)}, \boldsymbol{\theta}_k^{(i)})$ evaluates to 1. The algorithm hypothesizes future inputs of the system, $\hat{\mathbf{u}}_k$. In this work, we consider the situation where a single future input trajectory is known.

7. RESULTS

We performed a number of simulation-based experiments to analyze the performance of the prognostics approach using local damage estimation. For the purposes of comparison, we include results from the global estimation approach. For the local estimation approach, we use the five submodels derived in Section 4, where the three submodels associated with damage models use particle filters for joint state-parameter estimation, and the remaining two submodels, which require only state estimation, use extended Kalman filters. The global approach used $N = 500$, so when the three local particle filters each use $N = 167$, the total computational cost is equivalent to that of the global particle filter. We try also $N = 100$ and $N = 50$ to observe the changes in performance when less total computation is performed. Since the local estimators use measured values as inputs, performance will degrade as sensor noise is increased. We varied the sensor noise variance by factors of 1, 10, 100, and 1000, to explore this situation.

In a single experiment, combinations of wear parameter values were selected randomly within a range. We selected the true wear parameter values in $[1 \times 10^{-3}, 4 \times 10^{-3}]$ for w_A, and in $[1 \times 10^{-11}, 7 \times 10^{-11}]$ for w_t and w_r, such that the maximum wear rates corresponded to a minimum EOL of 20 hours. The local estimators had to estimate both the local states and the local unknown wear parameters. In all experiments, we used $T = 60\%$, $v_0^* = 50\%$, $v_\infty^* = 10\%$, and $P = 1 \times 10^{-4}$ for the variance control algorithm. We performed 20 experiments for each value of N and sensor noise level. We considered the case where the future input of the pump is known, and it is always operated at a constant RPM, in order to limit the uncertainty to only that involved in the noise terms and that introduced by the filtering algorithms.

The averaged estimation and prediction performance results are shown in Table 1. The part of the table with $|\mathcal{M}| = 1$ corresponds to results using the global model. The column labeled N lists the number of particles used per submodel, and the column labeled $\mathbf{n}$ lists the sensor noise variance multipliers. Here, we use percent root mean square error (PRMSE) as a measure of estimation accuracy, relative accuracy (RA) (Saxena, Celaya, Saha, Saha, & Goebel, 2010) as a measure of prediction accuracy, and RMAD as a measure of spread. Each are averaged over multiple prediction points for a single scenario (see (Saxena et al., 2010; Daigle & Goebel, 2011) for the mathematical definitions of the metrics used here). Note that all metrics are expressed as percentages.

We can see that in the case where $N = 167$, for the same amount of computation, the local estimation approach obtains results very close to the global approach for damage estimation accuracy. In some cases, performance is slightly better, and in other cases, slightly worse. At the highest noise level, the local approach improves significantly for estimation of w_A. This is mostly due to the convergence properties of the global approach. It tends to converge with much more difficulty than the local approach for the same amount of noise. RMADs of the wear parameters are also quite evenly matched. Here again, in some cases the local approach is slightly better, and in others slightly worse. As the sensor noise increases, the variance is naturally larger and more difficult to control, resulting in the increase in RMAD as sen-

Table 1. Estimation and Prediction Performance

$\|\mathcal{M}\|$	N	$\mathbf{n}$	PRMSE$_{w_A}$	PRMSE$_{w_t}$	PRMSE$_{w_r}$	$\overline{\text{RMAD}}_{w_A}$	$\overline{\text{RMAD}}_{w_t}$	$\overline{\text{RMAD}}_{w_r}$	$\overline{\text{RA}}$	$\overline{\text{RMAD}}_{RUL}$
1	500	1	3.70	3.58	2.54	11.58	11.27	10.03	97.28	11.61
1	500	10	4.15	2.81	2.74	12.25	11.48	10.63	96.58	12.34
1	500	100	6.30	3.46	3.23	13.46	12.38	11.59	94.69	14.09
1	500	1000	12.93	6.25	5.29	13.92	12.99	12.64	79.37	15.32
5	167	1	3.19	2.61	2.88	12.26	10.85	10.76	96.61	12.01
5	167	10	3.66	2.90	3.56	12.69	11.09	11.85	95.28	13.32
5	167	100	4.44	3.39	3.78	13.05	11.78	12.56	93.17	14.57
5	167	1000	5.59	4.46	8.26	14.72	12.86	15.09	87.66	16.19
5	100	1	4.02	3.49	3.49	12.42	10.68	10.77	95.90	12.15
5	100	10	4.52	3.78	4.27	12.69	11.04	11.45	95.16	13.30
5	100	100	5.71	4.02	6.05	12.99	11.81	12.73	91.82	14.74
5	100	1000	9.06	4.76	6.91	13.83	12.15	13.92	79.90	16.40
5	50	1	5.66	4.98	5.19	12.33	10.41	10.39	94.59	12.39
5	50	10	6.12	4.92	6.29	12.41	10.71	11.21	93.44	12.99
5	50	100	7.43	6.08	8.24	12.94	11.16	11.91	90.05	14.19
5	50	1000	14.03	9.33	14.41	12.90	11.66	12.46	73.05	13.62

sor noise increases for both approaches. As the number of particles used in the local approach is reduced, accuracy decreases, as expected, but not significantly. The RMADs actually generally decrease as the number of particles is reduced, corresponding to increased precision at the cost of decreased accuracy.

Prediction performance corresponds to the change in damage estimation performance. More accurate damage estimates correspond to higher RA, and increases in spread of the wear parameters leads to increases of the spread of the RUL. For $N = 167$, the performance is slightly worse, but still comparable to the global estimation approach, even though some of the covariance information in the global state estimate was lost due to the decoupling. At the highest noise level, the local approach has significantly better accuracy. This is due to the relatively poor convergence behavior of the global approach, leading to inaccurate early predictions that bring down the averaged RA. As N decreases, so that less total computation is being performed, accuracy reduces, but so does spread. For $N = 100$, less computation is performed with only a small change in performance. As N is reduced further to 50, performance begins to degrade. Moreover, as sensor noise increases, the local approach can lose its advantage over the global approach, and this occurs when N is reduced to 50. For $N = 50$ and the nominal amount of sensor noise, comparable prognostics performance is achieved to the global approach with less than a third of the computation. We expect the benefits of the local approach to be more pronounced as the dimension of the state-parameter space increases.

8. CONCLUSIONS

In this paper, we developed a novel distributed damage estimation approach for model-based prognostics that is based on a formal framework for structural model decomposition. Using the concept of PCs, a system model is decomposed into a set of minimal submodels. A local damage estimation problem is defined for each submodel. Local state-parameter estimates obtained using an appropriate filter are merged into a global state-parameter estimate from which EOL predictions are computed. Results demonstrate that equivalent, or in some cases, better prognostics performance can be achieved using this methodology with less computation than a global approach. Further, the approach can be naturally distributed and therefore may serve as a fundamental aspect of a practical system-level prognostics approach.

The idea of using model decomposition to improve state and parameter estimation is not new. For example, subspace methods (Katayama, 2005) for system identification employ QR-factorization and singular-value decomposition (Overschee & Moor, 1996) for solving identification problems in large-dimension systems. These methods are numerically robust for linear systems. Recently, several extensions have been proposed that apply to nonlinear systems (e.g., (Westwick & Verhaegen, 1996)). However, methods to automatically derive the decomposition from the system model have not been addressed.

An approach for decomposing a system model into smaller hierarchically organized subsystems, called *dissents*, is described in (Williams & Millar, 1998). PCs are conceptually equivalent to dissents, and previous work applied PCs for model decomposition to generate a more robust and computationally simpler parameter estimation approach for fault identification (Bregon, Pulido, & Biswas, 2009). Simulation results in that case showed an improvement in estimation accuracy while having a faster convergence to true solutions. Similar work was proposed in (Roychoudhury et al., 2009) using a dynamic Bayesian network (DBN) modeling framework, in which an automatic approach for model decomposition into factors based on structural observability was developed for efficient state estimation and fault identification. This approach also obtained an improvement in state estimation efficiency without compromising estimation accuracy. The relation between both approaches has been established in (Alonso-Gonzalez, Moya, & Biswas, 2010), where DBNs are derived from PCs for the purposes of estimation.

In future work, we will investigate extensions to system-level and distributed prognostics. For one, the model decomposition algorithm suggests a sensor placement strategy to optimize the decomposition of a system-level model into independent component models. One need only place sensors at the inputs and outputs of components to ensure that component models may be decoupled. The submodels derived from the PC approach are minimal in that, for a given output, they contain only the subset of the model required to compute that output as a function of only inputs and other measured outputs. Nonminimal submodels may be formed by merging minimal submodels, and this may be desired in some cases, e.g., if it eliminates using some high-noise sensors as inputs. This forms part of a more generalized model decomposition framework under development. As described in the paper, distributed damage estimation is an essential part of a distributed model-based prognostics architecture. The computation associated with the prediction problem in our approach can be trivially distributed (via parallel EOL simulations), but in future work, we would like to develop a decomposition of the prediction problem into local prediction problems for a fully distributed prognostics architecture.

ACKNOWLEDGMENTS

Anibal Bregon's work has been partially supported by the Spanish MCI DPI2008-01996 and MCI TIN2009-11326 grants.

REFERENCES

Alonso-Gonzalez, C., Moya, N., & Biswas, G. (2010). Factoring dynamic Bayesian networks using possible conflicts. In *Proc. of the 21th International Workshop on Principles of Diagnosis* (p. 7-14). Portland, OR, USA.

Arulampalam, M. S., Maskell, S., Gordon, N., & Clapp, T. (2002). A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, *50*(2), 174–188.

Biswas, G., & Mahadevan, S. (2007, March). A Hierarchical model-based approach to systems health management. In *Proc. of the 2007 IEEE Aerospace Conference.*

Bregon, A., Pulido, B., & Biswas, G. (2009, Sep). Efficient on-line parameter estimation in TRANSCEND for nonlinear systems. In *Proc. of the Annual Conference of the Prognostics and Health Management Society 2009.* San Diego, USA.

Bregon, A., Pulido, B., Biswas, G., & Koutsoukos, X. (2009). Generating possible conflicts from bond graphs using temporal causal graphs. In *Proceedings of the 23rd European Conference on Modelling and Simulation* (p. 675-682). Madrid, Spain.

Daigle, M., & Goebel, K. (2010a, October). Improving computational efficiency of prediction in model-based prognostics using the unscented transform. In *Proc. of the Annual Conference of the Prognostics and Health Management Society 2010.*

Daigle, M., & Goebel, K. (2010b, March). Model-based prognostics under limited sensing. In *Proceedings of the 2010 IEEE Aerospace Conference.*

Daigle, M., & Goebel, K. (2011, March). Multiple damage progression paths in model-based prognostics. In *Proceedings of the 2011 IEEE Aerospace Conference.*

Doucet, A., Godsill, S., & Andrieu, C. (2000). On sequential Monte Carlo sampling methods for Bayesian filtering. *Statistics and Computing*, *10*, 197–208.

Hutchings, I. M. (1992). *Tribology: friction and wear of engineering materials*. CRC Press.

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

Kallesøe, C. (2005). *Fault detection and isolation in centrifugal pumps*. Unpublished doctoral dissertation, Aalborg University.

Katayama, T. (2005). *Subspace Methods for System Identification*. Springer.

Luo, J., Pattipati, K. R., Qiao, L., & Chigusa, S. (2008, September). Model-based prognostic techniques applied to a suspension system. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, *38*(5), 1156 -1168.

Lyshevski, S. E. (1999). *Electromechanical Systems, Electric Machines, and Applied Mechatronics*. CRC.

Overschee, P., & Moor, B. D. (1996). *Subspace Identification for Linear Systems*. Boston, MA, USA: Kluwer Academic Publishers.

Pulido, B., & Alonso-González, C. (2004, October). Possi-

ble Conflicts: a compilation technique for consistency-based diagnosis. *IEEE Trans. on Systems, Man, and Cybernetics. Part B: Cybernetics*, *34*(5), 2192-2206.

Roychoudhury, I., Biswas, G., & Koutsoukos, X. (2009, December). Factoring dynamic Bayesian networks based on structural observability. In *Proc. of the 48th IEEE Conference on Decision and Control* (p. 244-250).

Saha, B., & Goebel, K. (2009, September). 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.*

Saha, B., Saha, S., & Goebel, K. (2009). A distributed prognostic health management architecture. In *Proceedings of the 2009 Conference of the Society for Machinery Failure Prevention Technology.*

Saxena, A., Celaya, J., Saha, B., Saha, S., & Goebel, K. (2010). Metrics for offline evaluation of prognostic performance. *International Journal of Prognostics and Health Management*.

Staroswiecki, M., & Declerck, P. (1989, July). Analytical redundancy in nonlinear interconnected systems by means of structural analysis. In *IFAC Symp. on Advanced Information Processing in Automatic Control.*

Westwick, D., & Verhaegen, M. (1996). Identifying MIMO Wiener systems using subspace model identification methods. *Signal Processing*, *52*(2), 235 - 258.

Williams, B., & Millar, B. (1998). Decompositional model-based learning and its analogy to diagnosis. In *Proc. of the Fifteenth National Conference on Artificial Intelligence* (p. 197-204).