Battery Capacity Anomaly Detection and Data Fusion

John Weddington¹, Wuzhao Yan¹, Wanchun Dou², and Bin Zhang¹

¹Department of Electrical Engineering, University of South Carolina, Columbia, SC, USA
weddingj@email.sc.edu
wyang@email.sc.edu
zhangbin@eee.sc.edu

²Department of Computer Science & Technology, State Key Lab. For Novel Software Technology, Nanjing University, Nanjing, Jiangsu, 210093, P.R. China
douwc@nju.edu.cn

ABSTRACT

Anomaly detection is a critical enabling technique of PHM, especially in safety critical applications. In order for the PHM system to begin prediction of remaining useful life of a given system or component, the fault must be detected. This paper presents an integrated anomaly detection system for state-of-health of lithium-ion batteries. Two algorithms for state estimation and anomaly detection are used: the extended Kalman filter and the particle filter. A Dempster-Shafer Theory-based fusion approach is implemented to reduce the uncertainty of detection. The results on battery data show that the fusion improves the detection results significantly.

1. INTRODUCTION

There is an increasing need in engineered systems of all types for detection and handling of faults. Traditional engineering includes receiving a product or system specification, the design process, analyzing the design for accuracy, redesigning components that are inaccurate, testing the design to standard, final redesign for production, and validation and verification. In the ideal scenario, the engineering design process filters out discrepancies, incorporates all known data and techniques in an accurate and robust manner, considers all possible failure modes, and allows engineers to reasonably predict the total life of the engineered product or system. However, no matter how well the system is designed, manufactured, and maintained, faults may occur in any given system due to harsh environment, manufacturing flaws, and excessive operational conditions. Current industry trends continue to drive the need for advanced understanding of engineered systems, and appropriate warning when the functionality of these systems deteriorates. Even if the engineering process has incorporated fault-tolerant design practices, it is important to implement fault-tolerant design practices, it is important to implement diagnostic systems into the engineered system.

Diagnosis consists of three parts: detecting, isolating, and identifying faults in order for the end user to make the best decision for the intended application. There are both model-based and data-driven diagnostic approaches. First-principles models require comprehensive system analysis to develop an accurate model, which is difficult for complex systems. Data-driven approaches, on the other hand, rely on large amounts of data that are generated with fault modes of interest and are statistically sufficient to design diagnostic systems (Wu, Zhu, Ge, & Zhao, 2015). Neuro-fuzzy methods have become popular in recent years for modeling systems, but the limitation for many applications is that no sufficient fault data is available for modeling (Viharos & Kis, 2014). This paper addresses the detection piece of diagnosis, and does not attempt to isolate or identify the fault type. The algorithms presented flag the user when an anomaly is detected.

On-board lithium-ion (Li-ion) battery detection involves the use of state estimation techniques (Orchard, Hevia-Koch, Zhang, & Tang, 2013; Zhang, Tang, DeCastro, Roemer, & Goebel, 2014). In the framework of Bayesian theory, state estimators are used to make one-step predictions of the current fault dimension, which is then filtered when the new measurement becomes available to obtain the posterior fault dimension distribution. The posteriori distribution is then compared with a baseline distribution (established from the data of healthy systems) to calculate the probability of detection. There are a number of different state estimators including wavelet analysis (Kim & Cho, 2015), Kalman filter (KF) (Gadsden & Habibi, 2011), multiple model adaptive estimation (Singh, Izadian, & Anwar, 2013; Wu, et al., 2015), extended Kalman filter (EKF) (Sidhu, Izadian, & Anwar, 2015), particle filter (PF) (Chen, Zhang, Vachtsevanos, Orchard, 2011; Chen, Brown, Sconyers, Zhang,
Vachtsevanos, Orchard, 2012; Goebel, Saha, Saxena, Celaya, & Christophersen, 2008), autoregressive integrated moving average (Box, Jenkins, & Reinsel, 2008; Saha, Goebel, & Christophersen, 2009).

After a fault is detected, prognosis is executed to estimate the time to failure. Each estimator differs from the other by offering better performance in different aspects. For example, the EKF is founded upon the efficient, optimal KF, which is able to accurately produce the underlying state for linear systems. The PF is a complex sampling algorithm that discovers the underlying state through many calculations, but it is well-suited for nonlinear systems. In order to utilize the best from each algorithm their results must be fused in some manner. Uncertainty management is a major consideration in real-world, online systems, especially in safety-critical applications. Each state estimation technique rolls into its estimate an undesirable level of uncertainty, and if fused improperly, the uncertainty could undermine the usefulness of the estimate.

This paper expands upon these concepts by combining the results from multiple state estimators and applying anomaly detection (AD) to the fused battery capacity estimation. First a fault dynamic model is developed. Using this model, EKF linearization equations can be created and integrated into the framework, along with the EKF and the PF estimation algorithms. As the data is passed into the algorithm, it passes separately through the EKF and the PF. Probability density functions (pdfs) are generated for each state estimation and are each fed into the Dempster-Shafer (DS) fusion algorithm. DS theory is well-suited to combine multiple sources in a neat framework. As will be demonstrated, the fusion of each state estimation results in tighter confidence bounds and reduced uncertainty, which is desirable in safety-critical applications.

The paper is organized as follows: The approach is developed from prior literature in section II. Section III details the implementation and experiments. Sections IV and V summarize the results and suggest expansion to the implementations described in this paper, respectively.

2. APPROACH DEVELOPMENT

Batteries are temporary power sources with limited lifespan. Over time, they experience degraded performance as a result of normal use, as well as accelerated degradation under strained operating conditions. This section details a method for AD of Li-ion batteries by combining the EKF and the PF to reduce the uncertainty inherent in each state estimator. Figure 1 demonstrates the flow chart utilized in designing the proposed AD system. Online fault detection is performed in a recursive manner by streaming in each sample, running each sample through the EKF and the PF separately, and then fusing the results from each algorithm. AD is performed on the fused result and a decision is made declaring either a healthy or a faulty battery. The AD algorithm used is a standard detection algorithm that flags anomalies when the baseline distribution and real-time fault state distribution are deviated a specified percentage. This paper will focus on three key portions of the AD design: modeling, algorithm development (EKF and PF), and data fusion.

![Flowchart for fused estimation and AD](image)

2.1. Extended Kalman Filter

For state estimation, the KF is known as a linear quadratic optimal filter. This algorithm is a linear, discrete time, finite dimensional time-varying state estimator that minimizes the mean-squared error (MSE) (Ribeiro, 2004). Capacity degradation of Li-ion batteries is a non-linear process, for which KF is insufficient and necessitates the use of EKF. The EKF expands the KF to incorporate non-linear processes, by linearizing around the current state, as described by the mean and covariance. The nonlinear process model (from cycle $k$ to cycle $k+1$) is described as a hidden Markov model (HMM) by

$$x_{k+1} = f(x_k, u_k) + w_k$$

where $x_k$ and $x_{k+1}$ are the features (vector) at the current time instant, $k$, and the next time instant, $k+1$, $f$ is a fault growth model, $u_k$ is the operating condition, and $w_k$ is a zero-mean Gaussian noise. The observation model is

$$z_k = h(x_k) + v_k$$

where $z$ is the observation vector, $h$ is a nonlinear observation function, and $v_k$ is the zero-mean Gaussian noise (Huang, 2010). EKF is evaluated in two stages: prediction and update. Both the prediction and update steps require the calculation...
of the partial derivatives (the Jacobian) of \( f(F_i) \) and \( h(H_i) \), as shown below:

State Jacobian: \( F_i = \frac{\partial f}{\partial x} \bigg|_{x_i} \)

(3)

Observation Jacobian: \( H_i = \frac{\partial h}{\partial x} \bigg|_{x_i} \)

(4)

The final equations for the EKF prediction and update steps after derivation are

State Predication: \( \hat{x}_{k+1|k} = f(\hat{x}_{k|k}, u_{k-1}) \)

(5)

Prediction Covariance: \( P_{k+1|k} = F_k P_{k|k} F_k^T + Q_k \)

(6)

State Update: \( \hat{x}_{k+1|k+1} = \hat{x}_{k+1|k} + K_k \left( z_k - h(\hat{x}_{k+1|k}) \right) \)

(7)

Innovation Covariance: \( P_{k+1|k+1} = (I - K_{k+1} H_{k+1}) P_{k+1|k} \)

(8)

Kalman Gain: \( K_{k+1} = P_{k+1|k} H_{k+1}^T (H_{k+1} P_{k+1|k} H_{k+1}^T + R_{k+1})^{-1} \)

(9)

2.2. Particle Filter

One disadvantage of the EKF is that the desired pdf is estimated by a Gaussian, and as the system under study is not normally distributed, the estimation could deviate from the actual distribution and diverge (Goebel, et al., 2008; Zhang, Khawaja, Patrick, & Vachtsevanos, 2010; Zhang, Sconyers, Byington, Patrick, Orchard, & Vachtsevanos, 2011). The PF is developed as a solution for nonlinear random non-Gaussian systems. The algorithm assumes the process state equations can be effectively modeled as a first-order nonlinear Markov process in Eqs. (1) and (2). The state \( x \) and observation \( z \) for cycles 1 to \( k \) are defined as

\[ x_{0:k} \triangleq \{ x_0, x_1, \ldots, x_k \}, z_{1:k} \triangleq \{ z_1, z_2, \ldots, z_k \} \]

(10)

The PF is also known as a sequential Monte Carlo method for state-space inference. PF is able to accommodate nonlinearities easily, provided enough particles are used. The particle filter uses a set of weighted particles to estimate the current and future state based upon a nonlinear fault dynamic model. The algorithm begins with a set of \( N \) particles available at cycle \( k - 1 \) sampled from the target distribution \( \pi_k \), as defined below,

\[ \{ x_{0:k-1}^{(i)} \}_{i=1,\ldots,N} \sim \pi_k(x_{0:k}) \]

(11)

where the target distribution is defined as the \( a \ posteriori \) distribution of \( x_{0:k} \) in Bayesian filtering,

\[ \pi_k(x_{0:k}) = p(x_{0:k} | z_{1:k}) \]

(12)

The objective of filtering is to obtain a set of \( N \) new particles and this set of particles is distributed according to the target distribution at cycle \( k \). To obtain these new particles, a known stationary distribution is chosen by the user to be the proposal or importance distribution. A set of \( N \) particles are selected from the proposal distribution, as shown,

\[ \{ x_{0:k-1}^{(i)} \}_{i=1,\ldots,N} \sim q_k(x_{0:k}) \]

(13)

and compared against the target distribution. The true distribution is approximated by a set of \( N \) weighted particles,

\[ \sum_{i=1}^{N} w_k^{(i)} = \delta \int q_k(x_{0:k}) \pi_k(x_{0:k}) dx_k \]

(14)

The sum of these weights is equal to 1. To get the consistent estimate of posterior distribution, importance sampling corrects the difference between \( q_k \) and \( \pi_k \) by setting weighting factors for each particle, which is given by:

\[ w_k^{(i)} = \frac{p(x_{0:k}^{(i)} | z_{1:k})}{q_k(x_{0:k}^{(i)})} \]

(15)

and is normalized as

\[ w(x_{0:k}^{(i)}) = \frac{w_k^{(i)}}{\sum w_k^{(i)}} \]

(16)

With this new set of weights, the target distribution can be approximated as:

\[ \pi(x_{0:k}) = \sum_{i=1}^{N} w_k^{(i)} \delta(x_{0:k} - x_{0:k}^{(i)}) \]

(17)

In a simple case of particle filter, Bootstrap filter, the importance density function is set as the a priori pdf,

\[ q_k(x_{0:k} | x_{0:k-1}) = p(x_{0:k} | x_{0:k-1}) \]

(18)

In this setting, the weights for the newly generated particles are proportional to the likelihood of new observations, i.e.

\[ w_k^{(i)} = w_k^{(i)} \cdot p(z_k | x_{0:k}^{(i)}) = w_k^{(i)} \cdot p(z_k | x_k) \]

(19)
In particle filters, degeneracy is a problem that must be addressed. Degeneracy can be described as the decreasing number of more heavily weighted particles as sampling continues to be performed. This leads to a dominance of particles with small weights describing the distribution. In practice, this results in inaccurate estimation of the actual state. Degeneracy is addressed by resampling. Resampling effectively replaces the smaller-weighted particles with larger-weighted particles, so as to describe to true distribution with higher veracity (Arulampalam, Maskell, Gordon, & Clapp, 2002).

Sequential Importance Resampling (SIR) is the PF implementation chosen for this application due to its robustness. The variance of the weighted particles generated by the Bootstrap filter is calculated using an effective sample size:

\[
\hat{N}_{\text{eff}} = \frac{1}{\sum_{i=1}^{N} (w_i^{(t)})^2}
\]  

(20)

When \(\hat{N}_{\text{eff}} < N_{\text{threshold}}\), the particles are resampled to eliminate particles with small weights. The steps in SIR are included in Figure 2.

**Sequential Importance Resampling (SIR) Steps**

- **Step 1**: Compute \(N^o = \left[ N \cdot w_i \right] \), this is, for each particle take the integrate part of the product. This identifies the particles with good weights and the number of times it should be kept in new samples.
- **Step 2**: Compute \(N = N - \sum N^o\), which indicates the number of particles to be resampled.
- **Step 3**: Compute \(N_{res} = \frac{N \cdot w_i^{(t)} - N^{(t)}}{\bar{N}}\) and compute its cumulative sum.
- **Step 4**: Generate \(\bar{N}\) random numbers within [0 1] and the range of these numbers that belong to \(N_{res}\).
- **Step 5**: The index of the range corresponds to the particle to be created.
- **Step 6**: Obtain the resampling index and resample the particles from \(q_k(x_{i,k} | x_{0:k-1})\).

**Figure 2.** Steps in SIR for PF.

### 2.3. Data Fusion

Data fusion is the process of combining the reasoning from different algorithms to achieve a result with high reliability. There are four prevalent methods for decision fusion: voting logic, possibility theory (fuzzy logic inference), probability theory (Bayesian fusion), and belief theory (DS-based fusion). Voting fusion is used heavily in industry with redundant instrumentation. Typical methods for voting fusion are threshold, majority, median, adaptive, inexact, and others (Parhami, 2005). Voting fusion is mostly used with crisp numbers, but its application has expanded into numerous domains, including dependable system design with unreliable components (Parhami, 2005).

Fuzzy logic inference expands voting into the domain of “gradual decision making” (Blank, Föhst, & Berns, 2010). Fuzzy logic is also called soft voting, because the fusion rules are not crisp, but allow for many different scenarios. The task of fusion using fuzzy methods was demonstrated in (Russo & Ramponi, 1994). Depending upon the complexity of the problem, fuzzy logic may require extensive understanding of system dynamics to perform effectively.

Bayesian theory is based on probability theory or Bayes’ theorem. The theory relates evidence and belief using prior and current information. Being a proven method, it maintains a larger contingent of supported applications of the theory. This is due, in part, to its simple formulation, which enables it to be better understood (Challa & Koks, 2004).

DS builds upon Bayes’ theory by incorporating unknowns and confidence measures into its calculations. These “degrees of belief” lead to classifying DS under the heading “Belief Theory” or “Evidence Theory” (Carl, 2001). The DS has certain advantages over the Bayesian method under certain circumstance (Shafer, 1976; Shafer, 1990). It is based on the concept of assigning a degree of belief or confidence to certain propositions on the basis of combining all the available evidence. DS accommodates uncertainty by mapping a proposition into an internal value in [0,1]. The end points of the interval can be interpreted as the lower and upper probability functions. The fundamental equation in DS for belief is

\[
(m_i \oplus m_j)(C) = \frac{\sum_{A \cap B \cap C} m_i(A)m_j(B)}{1 - \sum_{A \cap B \neq C} m_i(A)m_j(B)}
\]  

(21)

where \(C\) is the output, \(A\) and \(B\) are hypotheses, and \(m_i\) and \(m_j\) are the masses for the hypotheses (Wu, Siegel, Stiefelhagen, & Yang, 2002). The sum of the masses in the numerator can be known as the belief measure in the hypothesis, while the denominator is called the plausibility measure. The belief and the plausibility provide confidence bounds, which, when combined as indicated in Eq. (21), supply degrees of belief for the output. The DS method for data fusion is selected for this paper due to its ability to incorporate unknown system dynamics in a flexible framework.

### 3. Implementation

In this section, the proposed approach will be demonstrated in a case study of the capacity degradation of a Li-ion battery. The battery is a safety-critical component that provides power to system functions including command, control, communications, computers, and intelligence. Li-ion batteries are widely used due to the advantages in higher...
energy density, longer cycle life, no memory effect, and lower weight (Saha & Goebel, 2009). Since the life and state of the batteries are not directly measurable, state estimation techniques play an important role in estimating the battery state-of-health and state-of-charge.

In this implementation, the state-of-health of a Li-ion battery with rated capacity of 1.1Ah is used to verify the proposed approach. The charge-discharge cycle of the battery is conducted with the Arbin BT2000 system under room temperature at a discharge current of 1.1A. The charging and discharging of the battery are halted at the given cutoff voltage. The capacity degradation curve versus charge-discharge cycle is obtained by Coulomb counting.

3.1. Model Development

The data used for model development and testing were drawn from (He, Williard, Osterman, & Pecht, 2011). As battery charge-discharge cycle count increases, the battery capacity slowly degrades from baseline in an exponential form. At some point, the fade-rate increases rapidly. Figure 3 shows the capacity degradation of four Li-ion batteries.

![Battery Capacity Test Data](image)

Figure 3. Battery capacity test data, along with preliminary models and final model.

The model used in Figure 3 was developed based upon the 4 data sets. The model was manually tuned using an iterative process. The fault dynamic model has the following form:

$$\beta(k) = p_1(p_2 + p_3k + p_4k^n + p_5k^{n_1} + p_6k^{n_2})$$

where \( p = \{ p_1, p_2, p_3,..., p_{10} \} = [2.7e-3, 1e-7, 1e-4, 8e-8, 2.2, 8e-11, 3.25, 0.9, -0.11, 2.7] \) is the parameter vector, \( k \) is the time-index, and \( \beta \) is the model.

There are a variety of thresholds used in flag the current state as healthy or faulty. The baseline size is selected to be 50, and is the number of points in the feature that are considered standard, or in the case of battery health, the cycles that indicate a healthy battery. The false alarm rate is selected to be 5% in this implementation.

3.2. EKF Implementation

In order to implement the EKF, a proper model must be developed. The above section selects a basic polynomial regression model. This is suitable for our purposes and closely follows the fault mode of the battery capacity degradation. The following model for detection is augmented from the fault dynamic model in Eq. (22).

$$\begin{bmatrix} x_{d,1}(k+1) \\ x_{d,2}(k+1) \end{bmatrix} = f_k \begin{bmatrix} x_{d,1}(k) \\ x_{d,2}(k) \end{bmatrix} + n(k)$$

$$x_{d,1}(k+1) = x_{d,1}(k) + \beta(k) \cdot x_{d,2}(k) + \omega(k)$$

$$y(k) = x_{d,1}(k) + v(k)$$

$$f_k(x) = \begin{cases} [1 \ 0]^T \cdot \text{if } \left\| x - [0 \ 1]^T \right\| \leq \left\| x - [0 \ 1]^T \right\| \\
[0 \ 1]^T \cdot \text{else} \end{cases}$$

$$\begin{bmatrix} x_{d,1}(0) \\ x_{d,2}(0) \\ x_{d,0} \end{bmatrix} = [1 \ 0 \ 0]$$

where \( f_k \) is a nonlinear mapping, \( x_{d,1} \) and \( x_{d,2} \) are Boolean states that indicate normal and faulty conditions, respectively. \( y(k) \) is the battery capacity from Coulomb counting, \( \omega(k) \) and \( v(k) \) are noise signals, and \( n(k) \) is i.i.d. uniform white noise. The Jacobian is then calculated for each iteration of new data. As each new data point becomes available, it is analyzed using the EKF. The covariance parameters \( Q \) in Eq. (6) and \( R \) in Eq. (9) are used to adjust the dependence of the state estimate upon the model or upon the measurements. The state estimation pdf for EKF has a wide base, which translates to lower confidence in the predicted state. Figure 4 shows the pdf for the EKF algorithm at cycle 274. Cycle 274 is the latest cycle for AD without fusion, which will be discussed later. EKF is closer to an optimal filter because of its basis on the KF, which allows it to be a steadier filtering state estimator. The baseline pdf is shown in green. The yellow line is the AD threshold, and represents the 95% of the baseline pdf. Once 95% of the estimation pdf crosses the AD threshold, the algorithm flags the detection of a fault.

3.3. PF Implementation

As with the EKF, the PF implementation also requires the use of a model. Surrounding the model is the use of various parameters such as the number of particles used to estimate the true state. The larger the number of particles used in PF, the better the estimation. The choice of particle number, also known as sampling size, is a trade-off between computational efficiency and accuracy. Clearly, in offline applications,
computational efficiency is not as critical as it is to online systems. Our primary interest in implementation in embedded, decentralized architecture, so a trial was run to help select the min particles that can be used to achieve a high-performing PF. Figure 5 shows the integrated absolute error (IAE) for each iteration of particle number. The elbow point on the graph lies roughly at about 100 particles, which should provide sufficient performance and reduce the computational complexity.

Figure 5. Demonstration of the relationship between error and particles used for state estimation.

3.4. DS Fusion Implementation

DS fusion is implemented to integrate the mass density of EKF and PF state estimations. In Bayesian-based fusion techniques, typically two possible hypotheses are considered, which are that a fault is present ($\mu_1$) or a fault is not present ($\mu_2$). DS theory includes a third hypothesis which is that a fault is either present or not present ($\mu_3$), which signifies the uncertainty. The combination table for mass function is shown in Table 1. Using the baseline pdf, certain threshold values are calculated to assign mass functions for EKF ($m_1$) and PF ($m_2$) to each hypothesis. The detection threshold for capacity depletion, based upon the baseline data, is selected to be $0.9448 \pm 0.0054$ Ah. This correlates to healthy data above $0.9502$ Ah (20% of the baseline pdf), faulty data below $0.9394$ Ah (5% of the baseline pdf), and uncertain data in between.

Table 1. DS Mass Function Combination Rule.

<table>
<thead>
<tr>
<th></th>
<th>PF</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$m_1(\mu_1)$</td>
</tr>
<tr>
<td>E</td>
<td>$m_1(\mu_1)$</td>
</tr>
<tr>
<td>K</td>
<td>$m_2(\mu_2)$</td>
</tr>
<tr>
<td>F</td>
<td>$m_3(\mu_3)$</td>
</tr>
</tbody>
</table>

The combined probability of fault is computed using Eq. (21) by summing the mass function combinations that indicate a fault is present (shaded in yellow), and dividing by the conflicting hypotheses chosen to be the normalizing factor (shaded in blue). The uncertainty in the result is computed by summing the mass function combinations that indicate either hypothesis is possible (shaded in green), and dividing by the normalizing factor. The algorithm compares the masses of the PF and the EKF state estimation techniques,

Figure 6 shows the state estimate pdf for the PF algorithm. The confidence bounds are tighter than EKF, but the algorithm produces more sporadic state estimates, which tend to jump around. This effect is a result of degeneracy that must be corrected for. It is clear in Figure 6 that the PF estimation is more sensitive to shifts in the mean due to the tighter confidence bounds.

Figure 6. PF state estimate pdf at cycle 274.

![EKF Detection Distributions](image1.png)

![PF Detection Distributions](image2.png)
and uses the DS to calculate what the actual state is. Table 3 shows the probability of fault (PoF) and uncertainty (PoFU) for cycle 39 and cycle 80.

Table 2. Progression of PoF and PoFU.

<table>
<thead>
<tr>
<th>Estimate</th>
<th>Cycle 39</th>
<th>Cycle 80</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PoF</td>
<td>PoFU</td>
</tr>
<tr>
<td>PF</td>
<td>0.1101</td>
<td>0.2350</td>
</tr>
<tr>
<td>EKF</td>
<td>0.1795</td>
<td>0.4939</td>
</tr>
<tr>
<td>Fused</td>
<td>0.1374</td>
<td>0.1371</td>
</tr>
</tbody>
</table>

The fault detection system includes a parameter to determine the number of consecutive ADs before the program will declare that it has detected a fault. The number used in this implementation is three. While this is used to help mitigate false alarms, it also slightly reduces the sensitivity of the algorithm. As noted above, cycle 274 is the latest cycle where the algorithm detects a fault when DS fusion is not implemented. As shown in Table 3, the fusion algorithm consistently detects a fault with a high degree of confidence well before the AD performance algorithm. The time to failure (TTF) is the number of cycles from point of detection to total failure, when the capacity drops below 0.5 Ah. This attribute of DS fusion is an important finding from the experiment.

Table 3. Comparison of TTF of detection algorithm.

<table>
<thead>
<tr>
<th>Battery Data Set</th>
<th>TTF w/o Fusion</th>
<th>TTF w/Fusion</th>
<th>Early Warning</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS2_35</td>
<td>655</td>
<td>734</td>
<td>79</td>
</tr>
<tr>
<td>CS2_36</td>
<td>494</td>
<td>659</td>
<td>165</td>
</tr>
<tr>
<td>CS2_37</td>
<td>653</td>
<td>802</td>
<td>149</td>
</tr>
<tr>
<td>CS2_38</td>
<td>771</td>
<td>851</td>
<td>80</td>
</tr>
</tbody>
</table>

Runtime testing was performed to then demonstrate visually the effect of the fusion algorithm on the final state estimation pdf. Figure 7 shows how much tighter the confidence bounds are for DS fusion than for either PF or EKF. The reliability of the measurement is dramatically increased by fusing the results from both the EKF and the PF.

4. CONCLUSION

EKF is implemented to accommodate for nonlinear battery dynamics, whereas the PF is a Monte Carlo sampling method for state-space inference. By fusing the state estimations from each of these techniques, the AD algorithm was able to detect degradation more quickly and with a higher degree of confidence. Using the DS method for fusion successfully accounted for unknowns and performed very well over the entire feature set. Due to a limited feature set, this paper only demonstrates a model which was tested and trained on the same data. For better results, models should be trained on one set of data, cross validated on another set of data, and finally tested on a third set of data, to ensure that the model generalizes well to data it has never before encountered. The IAE metric helped to determine that using more particles in the PF implementation clearly results in higher degree of veracity in state estimation. However, IAE does not account for the fact that by definition, filtering algorithms are attempting to estimate the true state underlying the measured state. Measurements contain unknown variation and noise, and it is therefore not desirable for the algorithm to perfectly mimic the measurement, but instead to accurately trend about the measurement distribution mean.

5. FUTURE WORK

For future implementations, the issue of diagnosis-triggered prognosis should be tackled, using the fused result as its basis. Using an expanded dataset should allow algorithms to be both trained and cross-validated prior to final proof-testing. A better model should also be developed. Rather than strictly relying on an estimated battery capacity data-driven approach, it is advisable to incorporate battery physics into the model (Saha & Goebel, 2009). The algorithm is constructed to incorporate input and operating mode changes, such as those mentioned above, but data was unavailable to be incorporated for this application. Also, PF approaches suffer from high computational load when utilizing large particle sets for estimation. To alleviate the need for large particle sets, one might also consider the value of incorporating each state estimation’s “historically-estimated correctness” (Wu, et al., 2002) into the DS fusion algorithm. This would mean that as part of DS fusion of PF and EKF, prior performance of the algorithm should be incorporated in order to dynamically adjust the weight the fusion algorithm places on either the PF or the EKF estimation. The algorithm would consider how well PF or EKF has been tracking the true state, and using this data the fusion algorithm would place more trust in one or the other. This should enable a
smaller particle set to be used, while maintaining high integrity fusion.

ACKNOWLEDGEMENTS

The project is sponsored by the ASPIRE grant program at the University of South Carolina and the Woodrow W. Everett, Jr. SCEEE Development Fund in cooperation with the Southeastern Association of Electrical Engineering Department Heads.

REFERENCES


Sconyers, C., Byington, C., Patrick, R., & Vachtsevanos, G. (2011). A Probabilistic...