Complex & Intelligent Systems (2018) 4:271–282
https://doi.org/10.1007/s40747-018-0082-z

ORIGINAL ARTICLE

Towards online data-driven prognostics system

Hatem M. Elattar[1] · Hamdy K. Elminir[2] · A. M. Riad[3]

Received: 7 May 2018 / Accepted: 3 September 2018 / Published online: 10 September 2018

Abstract
Complex engineering systems are always working in harsh operating conditions. Internal wears and tears of such systems can lead to catastrophic failures which affect system operation and endanger human lives in many cases. To avoid sudden failures, continuous monitoring, fault diagnosis, and failure prognosis are required. Prognostics as a discipline plays the major rule in impending failure prevention. Offline prognostics system focuses on maintenance and logistics operations whereas online prognostics focuses on maintaining safe operation. In a previous work we successfully developed an offline prognostics system for aircraft turbofan engines' remaining useful life (RUL) estimation. In this paper we will show how we used the offline prognostics system as the baseline definition towards creating an online data-driven prognostics system to enable informed real-time decisions.

Keywords Complex engineering systems · Condition based maintenance · Online prognostics · Prognostics and health management · Soft computing

Introduction

OMPLEX engineering systems usually operate in harsh operational and environmental conditions. By the time, the system itself, subsystems, and components examine performance degradation. This degradation is either due to internal wears and tears during normal operation or due to fault progression. In either case sudden failure could happen which implies to total system loss and endangers human lives.

The main reason of sudden failures is that the diagnostics system cannot catch fault progression. To avoid such failures, the maintenance strategy needs to be changed from fail and fix to predict and prevent [1]. In other words, instead of being reactive to be proactive. Prognostics is the main driver to proactivity [2].

Prognostics can be defined as a remaining useful life (RUL) estimation process of system/subsystem/component [3, 4].

In the beginning of life (BoL), the system operates normally with its full health. When the system starts to degrade, this can be considered as a trigger point to the prognostics system. The prognostics system continues to operate and performs RUL estimation at subsequent prediction points Fig. 1.

Prognostics information can be used by the mission commander to perform the required actions such as mission replanning, resources allocation, and system reconfiguration. Figure 2 these actions can extend the RUL of the system to complete the mission safely [5]. In the same time, prognostics information is the main driver for condition-based maintenance (CBM) [6].

To develop a prognostics system for RUL estimation, one of the following methods can be used [7, 8]:

– Physics-based approach.
– Data-driven approach.

In the physics-based approach a physical model of the system is developed. This physical model is formed from mathematical equations that represent degradation phenomenon and failure modes. The physics-based prognostics requires complete domain knowledge which is hard to find especially for complex systems.

✉ Hatem M. Elattar
hatemelattar2@gmail.com

1 Information System Department, Faculty of Computers and Information Sciences, Mansoura University, Mansoura, Egypt

2 Faculty of Engineering, Department of Electrical Engineering, Kafr Elshiekh University, Kafr Elshiekh, Egypt

3 Faculty of Computers and Information Sciences, Mansoura University, Mansoura, Egypt

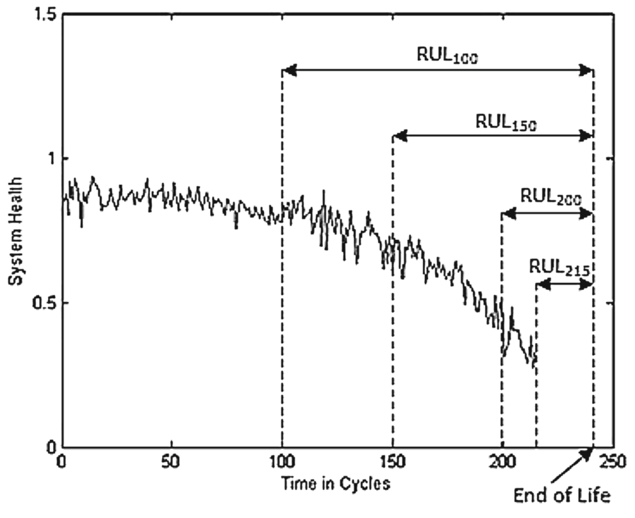


Fig. 1 RUL estimation at different prediction points

Data-driven prognostics uses historical data of performance and operational system parameters to create a model that links these parameters to system degradation for RUL estimation. Data-driven prognostics approach requires massive amount of historical data about system run in both normal and faulty modes of operations which is difficult to be obtained. But once the data is obtained, the creation of this model can be done easily.

Physics-based prognostics is preferred in online applications because of its high computational performance as well as prediction accuracy [9, 10]. However, it is too difficult, time consuming, and expensive to develop a physical model for complex systems. Even if the model is developed, it will not be accurate enough due to assumptions and simplification during the development process.

On the other hand, data-driven prognostics is cheap, accurate, and can be developed fast [3]. Although, it is computationally intensive which makes it suitable for offline prognostics applications.

Development of a robust data-driven prognostics system that deals with a huge amount of multivariate noisy data to perform accurate RUL estimation online is a big chall enge. But once this system is developed, it will make a paradigm shift in prognostics systems' development and deployment. In the same time, it will speed up prognostics maturation to increase its technology readiness level for onboard deployment as needed by the prognostics and health management (PHM) community [11].

In a previous work [12] we developed an offline prognostics system for RUL estimation of aircraft turbofan engines. In this study we complement the previous work to design, develop, and implement an online prognostics system based on data-driven approach.

RUL estimation using data-driven prognostics approach can be performed directly from the multivariate performance and operational parameters. Another way of data-driven methodology for RUL estimation is to predict system health state first then extrapolate/project the damage progression till end of life (EoL) [13]. We adopted the later way in our work because it is more descriptive from the engineering point of view.

Data analysis

The data used for both offline and online prognostics system developments is the turbofan engines' data from prognostics center of excellence data repository (PCoE), NASA [14]. The data is composed of two datasets: training and testing. Each dataset contains data about 218 turbofan engines' operational and performance parameters. Each engine in the training dataset has a complete run from BoL to EoL, whereas in test dataset the engine run is stopped before it reaches its EoL.

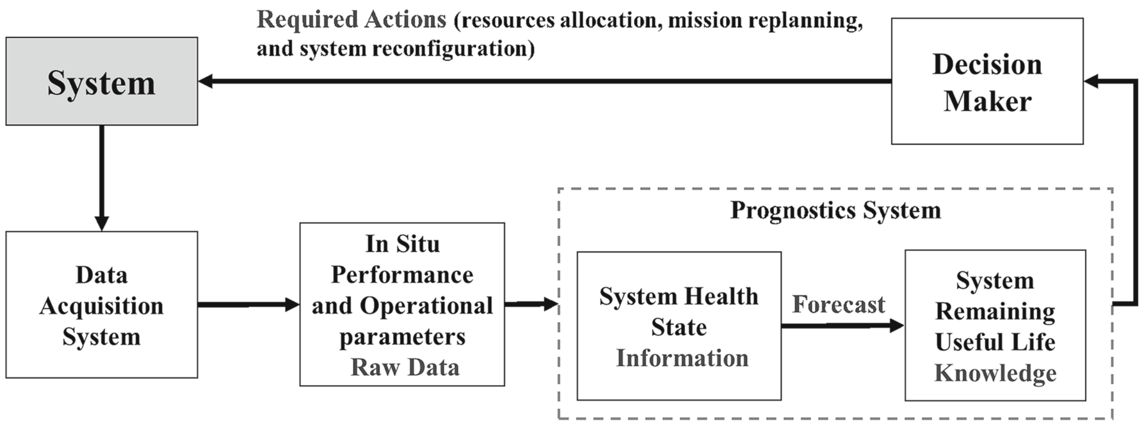
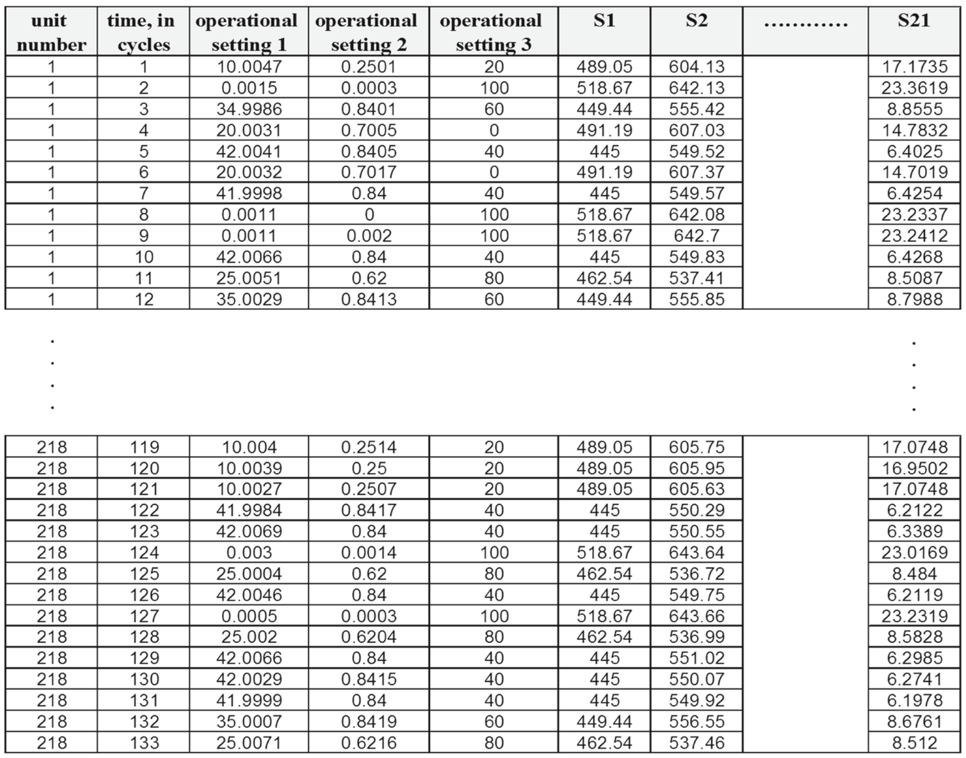

Required Actions (resources allocation, mission replanning, and system reconfiguration)
System
Decision Maker
Data Acquisition System
In Situ Performance and Operational parameters Raw Data
Prognostics System
System Health State Information
Forecast
System Remaining Useful Life Knowledge

Fig. 2 Prognostics in the decision-making process



unit number	time, in cycles	operational setting 1	operational setting 2	operational setting 3	S1	S2	…………	S21
1	1	10.0047	0.2501	20	489.05	604.13		17.1735
1	2	0.0015	0.0003	100	518.67	642.13		23.3619
1	3	34.9986	0.8401	60	449.44	555.42		8.8555
1	4	20.0031	0.7005	0	491.19	607.03		14.7832
1	5	42.0041	0.8405	40	445	549.52		6.4025
1	6	20.0032	0.7017	0	491.19	607.37		14.7019
1	7	41.9998	0.84	40	445	549.57		6.4254
1	8	0.0011	0	100	518.67	642.08		23.2337
1	9	0.0011	0.002	100	518.67	642.7		23.2412
1	10	42.0066	0.84	40	445	549.83		6.4268
1	11	25.0051	0.62	80	462.54	537.41		8.5087
1	12	35.0029	0.8413	60	449.44	555.85		8.7988
. . . .								. . . .
218	119	10.004	0.2514	20	489.05	605.75		17.0748
218	120	10.0039	0.25	20	489.05	605.95		16.9502
218	121	10.0027	0.2507	20	489.05	605.63		17.0748
218	122	41.9984	0.8417	40	445	550.29		6.2122
218	123	42.0069	0.84	40	445	550.55		6.3389
218	124	0.003	0.0014	100	518.67	643.64		23.0169
218	125	25.0004	0.62	80	462.54	536.72		8.484
218	126	42.0046	0.84	40	445	549.75		6.2119
218	127	0.0005	0.0003	100	518.67	643.66		23.2319
218	128	25.002	0.6204	80	462.54	536.99		8.5828
218	129	42.0066	0.84	40	445	551.02		6.2985
218	130	42.0029	0.8415	40	445	550.07		6.2741
218	131	41.9999	0.84	40	445	549.92		6.1978
218	132	35.0007	0.8419	60	449.44	556.55		8.6761
218	133	25.0071	0.6216	80	462.54	537.46		8.512

Fig. 3 Snapshot of the turbofan engines data

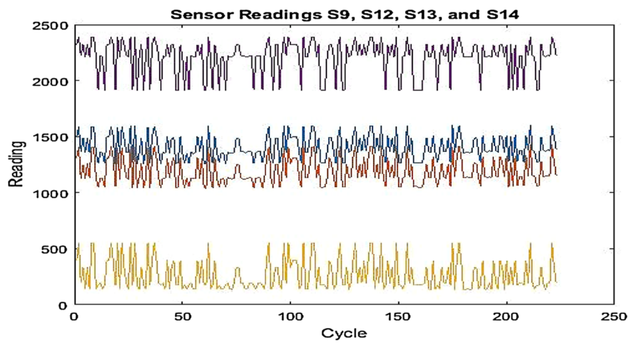


Fig. 4 Sensor measurements for sensors S9, S12, S13, and S14

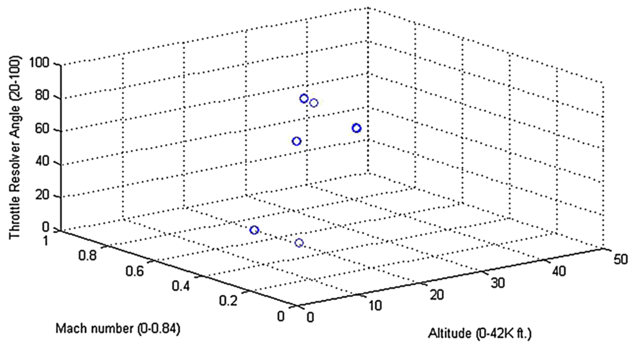


Fig. 5 Six operating regimes

The data composed of three-operational parameters and twenty-one performance parameters Fig. 3. There is no given index about health state (health index HI) so; it needs to be inferred from the aggregation of the given parameters.

Data analysis [12, 15–18] clarified that the data has the following characteristics:

- Operational parameters from sensors S1 through S21 are very noisy Fig. 4.
- Data is clustered into six operating regimes based on the three-operational settings Fig. 5.
- Sensor readings can be categorized into three types: monotonic readings, continuous inconsistent readings, and discrete readings Fig. 6.

Offline prognostics system

Offline prognostics systems are usually used in fleet wide health management. It can be considered as the main source of information to enable CBM instead of regular reactive or scheduled maintenance. Offline decisions are used for maintenance planning and logistic support. The offline prognostics systems do not have the time criticality and do not have to work in real-time [19]. So, it can be performed on computers with plenty of resources.

Data-driven prognostics is very suitable for offline applications which perform heavy computational tasks and do not have any computational resources restrictions.

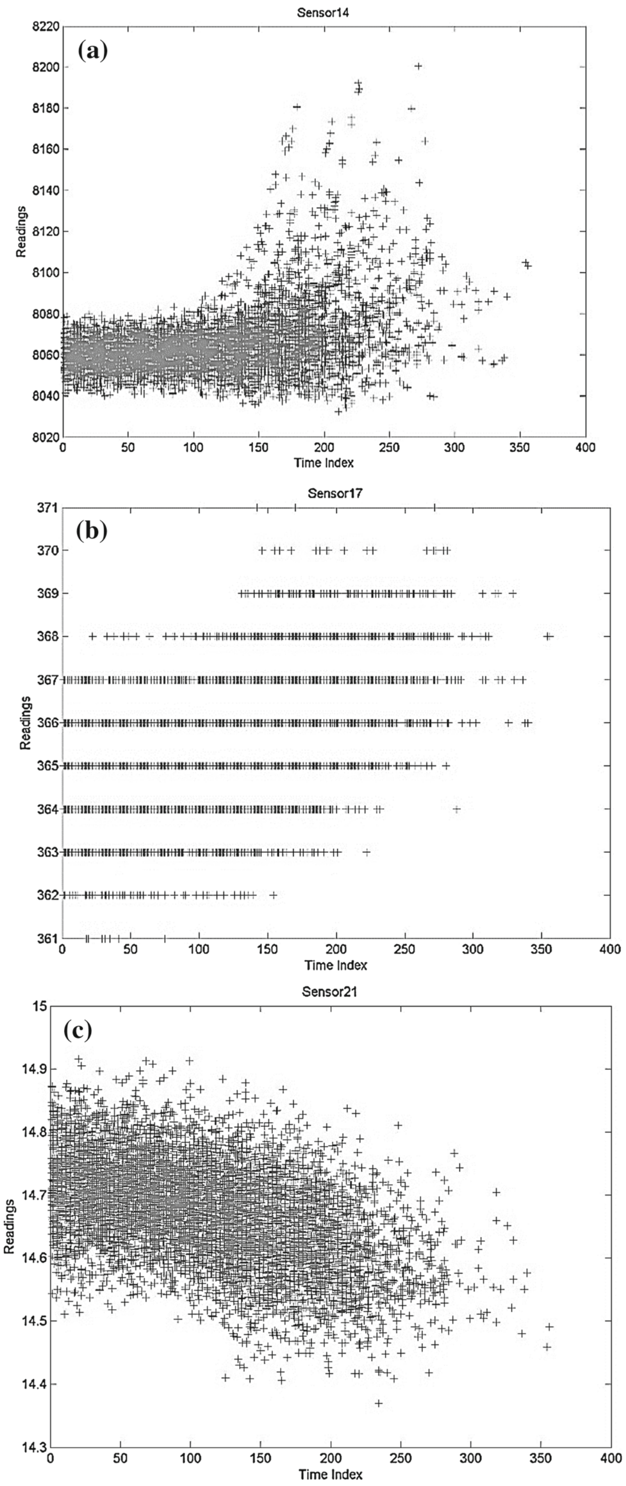


Fig. 6 Sensor readings, **a** continuous inconsistent, **b** discrete, **c** monotonic

In a previous study we developed an offline prognostics system for RUL estimation of aircraft turbofan engines [12].

The offline prognostics system Fig. 7. Takes the operational and performance parameters as an input to a well-trained neural network for HI prediction. The predicted health indices (HIs) are then smoothed by simple moving average technique. Polynomial curve fitting is performed using the smoothed HIs. Extrapolation of HI using the obtained polynomial equation is performed until HI reaches its minimum value (HI = 0). The RUL estimation is calculated by subtraction of the time cycle at the beginning of prediction from the time cycle at the EoL.

PCoE training dataset is used for system training whereas test dataset is used for system test and RUL estimation.

The offline prognostics system is divided into three parts: data preprocessing, neural networks training, and RUL estimation (testing).

Data preprocessing

In this part the training dataset is prepared for neural network training. The following tasks are performed:

– Features extraction for sensors that has monotonic trend S2, S3, S4, S7, S11, S12, S15, S20, and S21. Only these features are used for regression models development to infer HI. The other features are excluded.
– Performance assessment and data reduction are performed by setting HI = 1 for early cycles (fully healthy engine) in engine run and 0 for late cycles (failed engine) [20]. Other cycles are excluded from data. This creates a sample set that contains the nine selected sensor readings and the equivalent HI for the selected cycles.
– Data is partitioned into six operating regimes using operating regime partitioning based on the operational settings [21]. This creates six subsets each of them represents different operating regime.
– Six regression models are created based on linear least square method (1). Each model describes the relation between sensor readings and HI. Six features out of the nine are used because they show high correlation with HI (S2, S3, S4, S11, S12, and S15).

$$y = \propto + \sum_{i=1}^{N} \beta_i x_i + \varepsilon, \tag{1}$$

where y represents HI (dependent parameter), $x_1, x_2, \ldots, x_N$ are features for sensor readings (N dimensional vector for independent parameters), in our case $N = 6$, $(\alpha, \beta) = (\alpha, \beta_1, \beta_2, \ldots, \beta_N)$ are $N + 1$ model parameters, and ε is the noise term.
– The six regression models developed in the previous step are applied to every cycle in the training dataset (no data reduction) to infer HI.
– After HI inference, new features are extracted for neural network training. Operational settings, Monotonic sensors,

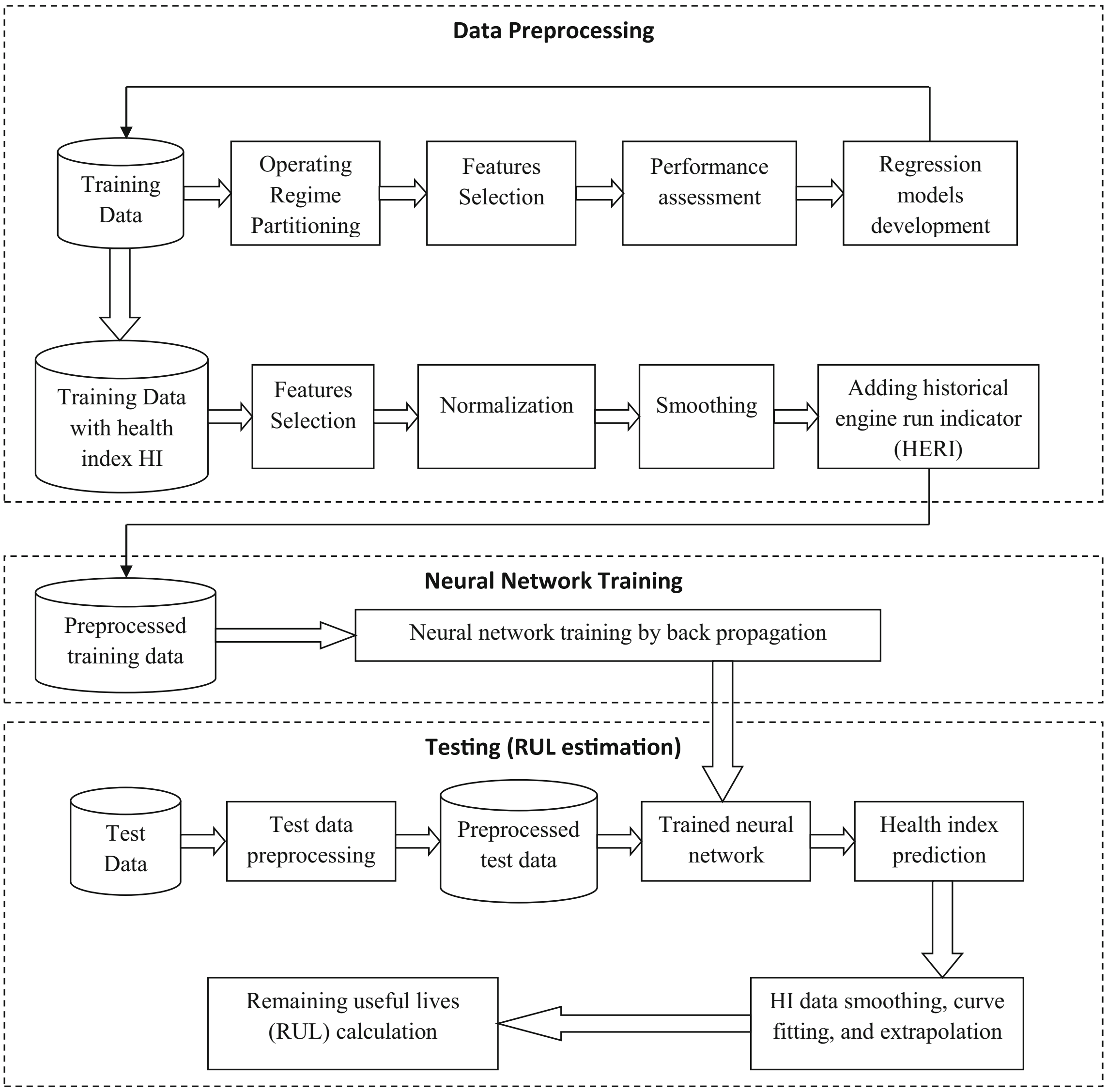

Data Preprocessing
Training Data
Operating Regime Partitioning
Features Selection
Performance assessment
Regression models development
Training Data with health index HI
Features Selection
Normalization
Smoothing
Adding historical engine run indicator (HERI)
Neural Network Training
Preprocessed training data
Neural network training by back propagation
Testing (RUL estimation)
Test Data
Test data preprocessing
Preprocessed test data
Trained neural network
Health index prediction
HI data smoothing, curve fitting, and extrapolation
Remaining useful lives (RUL) calculation

Fig. 7 Offline prognostics system

and continuous inconsistent features are selected (S2, S3, S4, S7, S8, S9, S11, S12, S13, S14, S15, S20, and S21).
– Data normalization and smoothing are performed in order to enhance neural network input to improve its performance.

Engines data is time series that requires dynamic neural network to deal with. Having said that, it is better to find a way to use static network to deal with dynamic data to simplify the system. To that end, six new features are invented. The six new features are called historical engine run indicators (HERIs) Fig. 8. HERIs are features account for number of runs for the engine in the operating regime. HERIs give information to the selected static multilayer perceptron neural network (MLPNN) about the history of engine run in the equivalent operating regime. This eliminates the need to use a dynamic network.

Neural network training

The input data to the MLPNN is composed of the three-operational settings, thirteen sensors, and the six HERIs. The target data are the inferred HI by the regression models.

The MLPNN has 22 inputs, 20 nodes in the only hidden layer with tan sigmoid transfer function, and single node in the output layer with linear transfer function. The network is trained by back propagation using "Levenberg—Marquardt" method. The best training results are: correlation coefficient.

$R = 0.98783$ and mean square error $\mathrm{MSE} = 0.00171$ after 30 epochs.

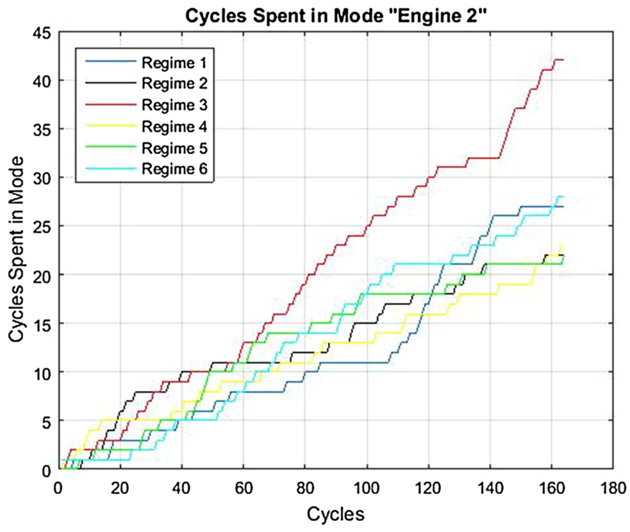


Fig. 8 HERI for engine 2 in the training dataset

Now we have a trained MLPNN that can predict HI from both operational and performance engines' parameters.

Remaining useful life estimation (testing)

In this part of the system, the testing dataset is used for RUL estimation. The same data preprocessing steps applied for the training data are also applied for the test dataset such as features extraction, normalization, sensor readings smoothing, and HERIs calculation.

The preprocessed data are inputted to the trained MLPNN to predict HI for each cycle for each engine in the test dataset. The predicted HIs by the MLPNN are noisy and need to be smoothed before curve fitting. The predicted HI data are smoothed using simple moving average. Polynomial curve fitting is applied for the smoothed HI. The obtained polynomial equations (one equation for each engine in the test dataset, i.e., 218 different polynomial equations, one for each engine) are used for HI curve extrapolation. The extrapolated curve intersects with the x-axis, i.e., when $\mathrm{HI} = 0$. The RUL is calculated by subtracting the cycle number at the prediction point $\mathrm{t_p}$ from cycle number at EoL t_{EoL} (2).

$$\mathrm{RUL} = t_{\mathrm{EoL}} - t_{\mathrm{p}} \tag{2}$$

Figure 9 shows the predicted HI by the MLPNN, the smoothed HI, and the polynomial fit that is used for extrapolation for engines 2 and 22 in the test dataset.

The RUL estimation for all engines in the test dataset is evaluated using the PHM08 evaluation function [4]. The score is compared to the top twenty scores in the PHM08 data challenge competition and can be placed in the fifteenth position.

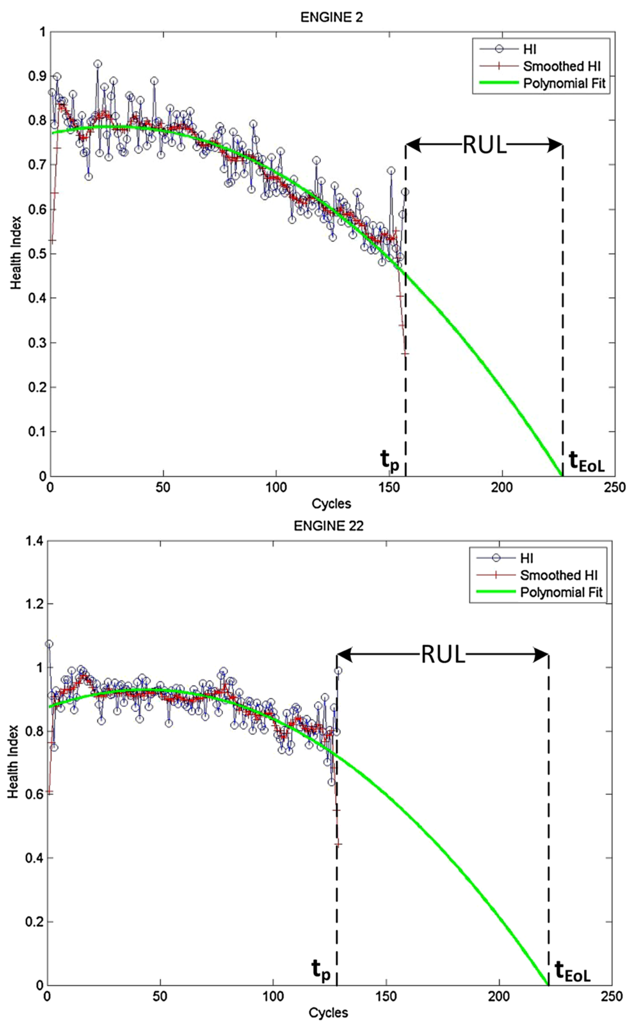


Fig. 9 Remaining useful life estimation

Results from the previously discussed offline prognostics system are great. However, it cannot be used for online applications for the following reasons:

- Computationally intensive and cannot be used with the limited onboard resources.
- Test dataset does not contain HI ground truth data that can be used for online RUL estimation evaluation.
- Techniques such as moving average, curve fitting, and extrapolation require massive amount of memory with respect to the available onboard memory.
- Online time criticality requires RUL estimation update each single or few cycles which is not the case in the offline applications.

In the following section we will show how did we benefit from the developed offline prognostics system and build a data-driven prognostics system that can be used online. This

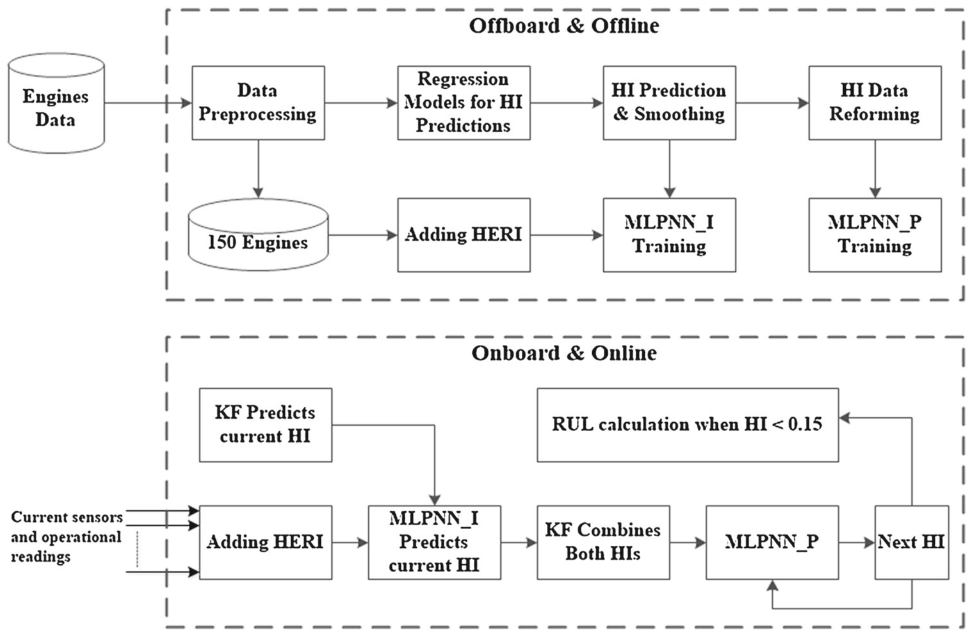


Fig. 10 Online prognostics system

prognostics system can be implemented onboard and works in real-time environment.

Online prognostics system

Online actions are time critical and must be taken in real-time [22]. Online prognostics systems give the required information to the mission commander about the anticipated failure time. The decision maker should do the required counter-actions to avoid failures and assure safe operation. Online prognostics systems must be robust, simple, work in real-time, implemented onboard, and do not consume too much computational resources.

To that end, we designed and developed an online prognostics system based on data-driven approach. In addition, we implemented this system on a single board computer to make sure of its online applicability.

The data used for system training and testing is the training dataset set from PCoE data [14]. The training dataset is subdivided into two subsets; one for system training (150 engines) and one for testing (68 engines).

System description

The online prognostics system Fig. 10. Is composed of two main modules offline and online modules. The offline module concerns about heavy computational tasks that cannot be performed onboard such as data preprocessing and system training. These tasks are not needed onboard because there is no online system training. The online part runs onboard and uses the trained techniques for RUL estimation in real-time. The online module is implemented on the single board computer.

As long as the decision is made for the system to be implemented onboard and run in real-time, the used techniques must be selected carefully and connected together in a way to achieve this goal.

To this end, the following is performed:

Static neural network is used instead of dynamic network for HI prediction with added HERIs to boost the computational process.

In the online system we do not use curve fitting and extrapolation as we did in offline system because they require storage of huge amount of data. In the same time, they are computationally intensive which cannot be done each cycle or few cycles as needed in online prognostics. Instead, we apply the projection concept using another static neural network which requires only 10 HI points to forecast the subsequent HI.

We replaced the HI smoothing method used in the offline system (simple moving average) by the recursive Kalman filter (KF) estimator [23]. KF is recommended for online applications because it’s recursive nature that does not require to save historical data in memory [24, 25]. We also benefit from KF to combine the HI prediction output from MLPNN with the KF model output and achieve more accurate HI prediction as we will discuss latter.

Offline (offboard) module

As mentioned above, in this module the heavy computational training tasks are performed. The output from this module is two static neural networks. We named these

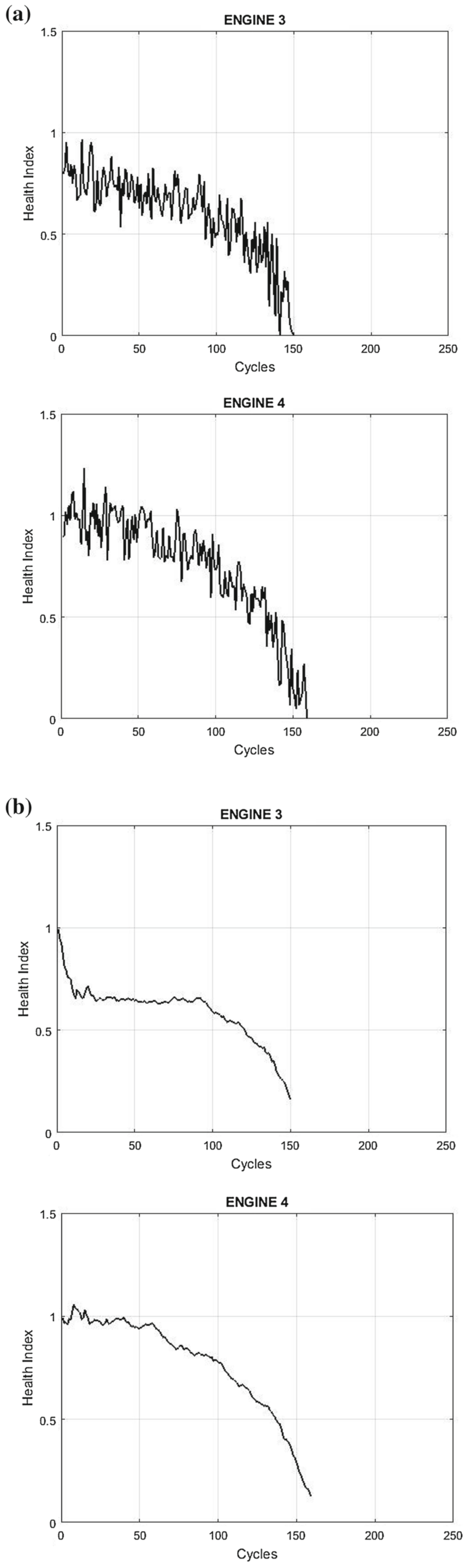


Fig. 11 Inferred HI ground truth data for 2 engines, **a** Unsmoothed HI, **b** Smoothed HI using KF

Table 1 Neural networks training results

Network	Epochs	MSE	*R*
MLPNN_I	244	0.0037712	0.97091
MLPNN_P	15	0.00011247	0.97091

networks as MLPNN_I and MLPNN_P. The MLPNN_I is trained to infer the current health state from aggregation of operational and performance parameters as well as HERIs based on the operating regime. The MLPNN_P is intended to use the last inferred HIs by MLPNN_I combined with KF (specifically last 10 HIs) and project them into future to forecast the subsequent HI.

To benefit from the predeveloped offline prognostics system, some of the used data preprocessing steps are applied to the training data. These steps are feature extraction, performance assessment, and operating regime partitioning as well as regression models development to calculate the HI ground truth data. Additionally, we applied KF to smooth the calculated HIs Fig. 11.

The final smoothed HI is used as follow:

- Target data for the MLPNN_I which its input data is the operational and performance parameters as well as HERIs.
- Reformed to be ten HIs as inputs and the eleventh as target data for MLPNN_P training.

After data preparation MLPNN_I and MLPNN_P are trained using “Levenberg–Marquardt” back propagation method.

The training results are shown in Table 1.

Online (onboard) module

This module is the system part (algorithm) that will be implemented onboard and should work in real-time. This module takes the sensor readings about engine performance and operating conditions from the data acquisition system. Based on the operational settings, the algorithm identified the operating regime. The HERI for the detected regime is increased by one. The operational and performance parameters as well as the six HERIs are imputed to the trained MLPNN_I. The MLPNN_I predicts the current system health state HI. In the same time, the KF predicts current HI based on its state estimation model. In the KF update step, it uses the HI calculated by MLPNN_I as a measurement to update the HI predicted by its model. Then, a corrected version of HI is calculated. This corrected HI has higher accuracy and precession than either HI calculated by KF state estimation model or MLPNN_I.

In the early part of system life, there is no need to do RUL estimation because the system does not suffer from

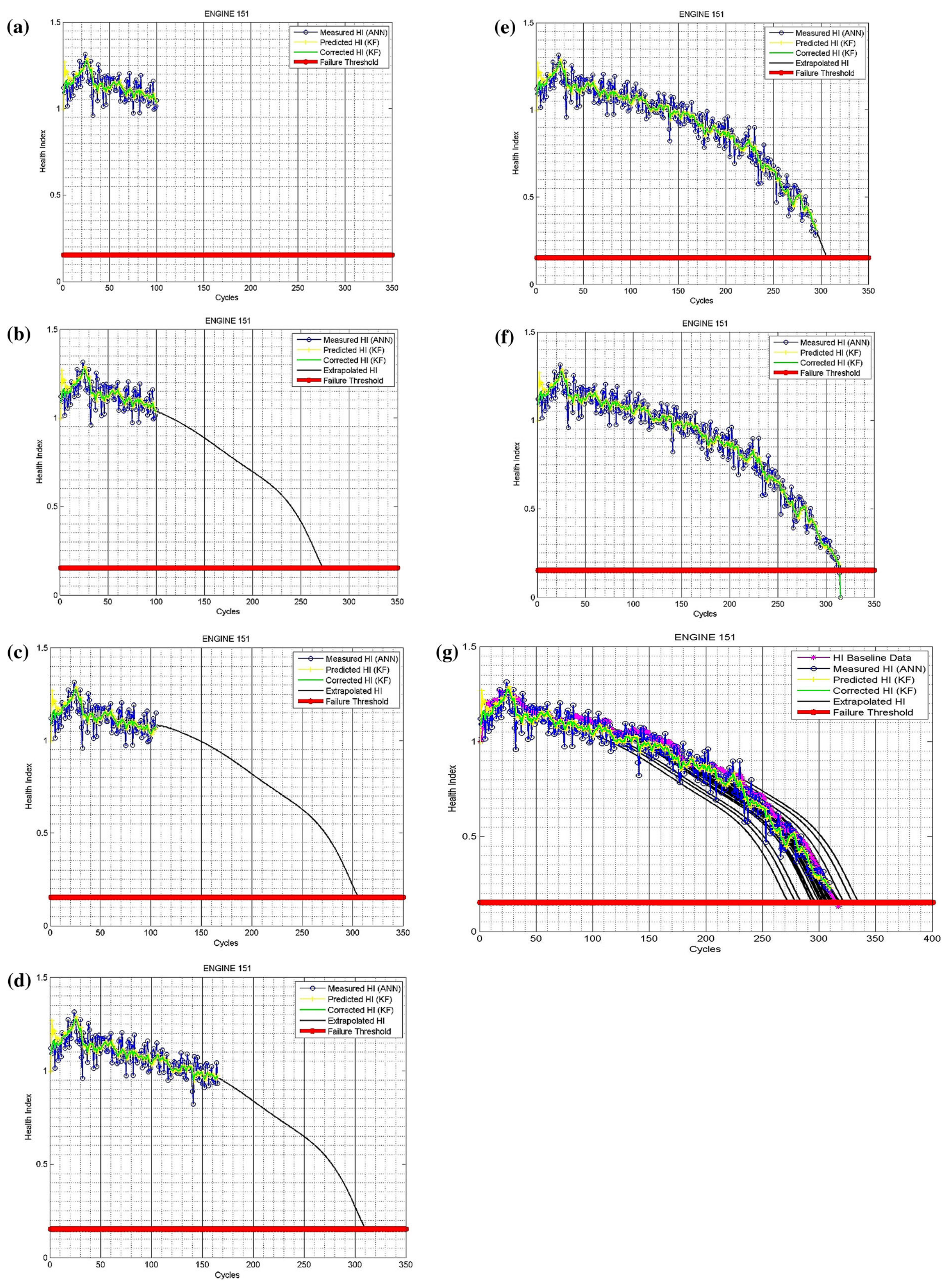


Fig. 12 RUL estimation at different time steps. **a** Output from KF and MLPNN_I before $t = 100$, **b** RUL estimation using MLPNN_P at $t = 100$, **c** RUL estimation at $t = 105$, **d** RUL estimation at $t = 165$, **e** RUL estimation at $t = 295$, **f** failure criteria is met and RUL estimation stopped, **g** RUL estimation at all time steps

Table 2 Online prognostics system performance evaluation

Prediction performance			
Mean squared error (MSE)		Average Bias	
Global	All engines	Global	All Engines
0.0043786	From 0.0017 to 0.0254	0.0217	From − 0.1124 to 0.1321
Computational performance			
RAM usage	Execution time (RUL = 173)	Execution time (RUL = 1)	Footprint
304 KBytes	7 ms	0.08 ms	80 KBytes

any degradation. As degradation starts the RUL estimation process must be instantly executed. In this study an arbitrary time point $t = 100$ is set at which RUL estimation starts. The selection of $t = 100$ is obtained from data analysis. Data analysis showed that most of engines starts to degrades in a point near $t = 100$.

When the system reaches the specified time point ($t = 100$), the RUL estimation starts. The MLPNN_P takes the last ten HIs to forecast the subsequent HI then get rid of the first HI from the previous ten. The MLPNN_P uses the new ten HIs to forecast the subsequent HI and again get rid of the first point. The process is repeated until HI reaches the failure criteria. The failure criteria in this case are $\text{HI} < 0.15$. The RUL is calculated by subtracting the time index at the prediction point from the time index when the engine fails and reaches the failure criteria.

To achieve real-time performance, the process from sensors data acquisition to RUL estimation must be completed before the next readings are acquired, i.e., in a time smaller than sensors polling time.

After RUL estimation, the system takes the new sensor readings and repeats the whole process to update the RUL estimation.

In this study engines are only monitored and RUL estimation is performed till EoL to test the prognostics system performance. In real settings, the decision maker should take the prognostics information (HI and RUL) to perform the required adjustment to avoid engine failure.

Figure 12 shows step by step operation of the online part of the system described above.

To prove that the system can work online on aircraft onboard computer, the online module should be implemented on a computer similar in specifications to the existing onboard computers. To this end, the online part is implemented on Raspberry Pi single board computer [26].

We consider our implementation to be platform independent and benefit from the internet of things (IoT) technology in the same time. So, the system is implemented as a universal windows platform (UWP) application.

Implementation of the prognostics system as a UWP application makes it easy to move across many devices such as tablets, PCs, mobiles, and single board computers.

The implementation is performed using Microsoft visual studio 2015 and coded in C++. The operating environment is Windows 10 IoT core which is a UWP used specifically for smaller and lower cost industry devices.

Results and discussion

To make sure that the system gives confident results and works in real-time, evaluation metrics must be applied. Mean squared error (MSE) and average bias (B) are used to evaluate RUL estimation results.

MSE and average bias evaluate the accuracy of the system results with respect to the ground truth data for each engine and globally for the whole engines. Average bias specifically has a very important meaning in prognostics applications. In prognostics, positively biased system is much preferred than negatively biased one. Positive bias means that the estimated RUL is less than the actual one so the decision can be made earlier. In contrast, negative bias is the opposite and can lead to failure before counteraction is taken.

Computational performance is also measured while the system runs onboard.

Global system and single engine RUL estimation MSE is governed by (3) and (4), respectively.

$$\text{MSE}(i) = \frac{1}{L}\sum_{l=1}^{L}\Delta^{l}(i)^{2}, \tag{3}$$

where L is the number of all engines, l is the number of the specific engine, $\Delta^{l}(i)$ is the error in RUL estimation for l^{th} engine at $t_{\text{p}} = i$, t_{p} is the time instant when the first RUL prediction is made, and $\Delta^{l}(i) = r_{*}^{l}(i) - r^{l}(i) * \ldots$ defines the notion of ground truth data.

$$\text{MSE}(l) = \frac{1}{\Delta t}\sum_{i=t_{\text{p}}}^{t_{\text{EOP}}}\Delta^{l}(i)^{2} \tag{4}$$

where t_{EOP} is the end of prediction time, $\Delta t = t_{\mathrm{EOP}} - \mathrm{t_p} + 1$

MSE range is (0, ∞), and the perfect result = 0.

Global system and single engine RUL estimation average bias is governed by (5) and (6), respectively.

$$B_i = \frac{\sum_{l=1}^{L}\{\Delta^l(i)\}}{L} \quad (5)$$

$$B_l = \frac{\sum_{i=t_{\mathrm{p}}}^{t_{\mathrm{EOP}}}\{\Delta^l(i)\}}{t} \quad (6)$$

The average bias range is (− ∞, ∞), and the perfect result = 0.

The system results evaluated for each engine selected for test as well as globally for all engines. System footprint, dynamic RAM usage, and execution time are measured to evaluate the computational performance. Results are shown in Table 2.

From the above results we can conclude the following:

– System prediction accuracy based on the used evaluation metrics (MSE and average bias) is very high globally and separately for each single engine.
– The results are very close to the perfect result = 0.
– System is positively biased which is preferred in prognostics application.
– System foot print and RAM usage are very small.
– Execution time shows that the system is calculating RUL a way faster than the polling time of sensors which is in common approximately 70 ms.
– Execution time proofs the real-time performance of the system. The other computational performances metrics shows the ability to implement this system onboard.

The above results proofs that the objective from this study is obtained which is to develop robust and accurate online prognostics system based on data-driven approach that can be implemented onboard and works in real-time settings.

Conclusion

In this study we intended to benefit from a predeveloped offline data-driven prognostics system and introduce an online prognostics system. The online prognostics system presented here is based on data-driven approach which is a big challenge because of the limited onboard resources.

Real-time performance, system complexity, computational resources reduction, as well as platform independent are all requirements and constraints faced during the proposed online prognostics system development lifecycle.

To deal with this challenge, static neural networks are suited to be used with time series data instead of dynamic networks and achieve high accuracy results with minimum onboard resources. A recursive estimator method (Kalman Filter) is used instead of normal filtration method for the same reason. Projection of HI instead of traditional mathematical extrapolation methods is used as well.

The output from the proposed online prognostics system is accurate and works in real-time which allows the decision maker to take informed onboard decisions on time to deal with fault progression and avoid impending failures to assure safe and reliable operation.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Voisin A, Levrat E, Cocheteux P, Iung B (2010) Generic prognosis model for proactive maintenance decision support application to pre-industrial emaintenance test bed. Intell Manuf 21(2):177–193. https://doi.org/10.1007/s10845-008-0196-z
2. Bond L (2008) Diagnostics and prognostics: state of the art and programs. In: PHM08 tutorial materials, IAEAWorkshop, Argentina, 10 Dec
3. Elattar H, Elminir H, Riad A (2016) Prognostics a literature review. Complex Intell Syst 2(2):125–154. https://doi.org/10.1007/s40747-016-0019-3
4. Saxena A, Goebel K, Simon D, Eklund N (2008) Damage propagation modeling for aircraft engine run-to-failure simulation. In: International conference on prognostics and health management, Marriott Tech Center Denver, CO, USA, pp 1:9, October 6–9. https://doi.org/10.1109/phm.2008.4711414
5. Elattar H, Elminir H, Riad A (2018) Conception and implementation of a data-driven prognostics algorithm for safety–critical systems. Soft Comput. https://doi.org/10.1007/s00500-017-2995-7
6. Imanian A, Modarres M (2017) A thermodynamic entropy-based damage assessment with applications to prognostics and health management. Struct Health Monit 17(2):240–254. https://doi.org/10.1177/1475921716689561
7. Karman J, Rafael G, Noureddine Z (2017) State of the art and taxonomy of prognostics approaches, trends of prognostics applications and open issues towards maturity at different technology readiness levels. Mech Syst Signal Process 94(214):236. https://doi.org/10.1016/j.ymssp.2017.01.050
8. Schwabacher M, Goebel K (2007) A survey of artificial intelligence for prognostics. In: Proceedings of AAAI fall symposium, November 9–11, Arlington, VA
9. Balaban E, Saxena A, Narasimhan S, Roychoudhury I, Koopmans M, Ott C, Goebel K (2015) Prognostic health-management system development for electromechanical actuators. Aerosp Inform Syst 12(3):329–344. https://doi.org/10.2514/1.I010171

10. Medjahar K, Zerhouni N (2009) Residual-based failure prognostic in dynamic system, 7th IFAC international symposium on fault detection, supervision and safety of technical processes, June 30–July 3. Sants Hotel, Spain
11. Feather M, Hicks K, Mackey R, Uckun S (2008) Guiding technology deployment decisions using a quantitative requirements analysis technique. In:16th IEEE international conference on requirements engineering, September 8–12, Barcelona, Spain. https://doi.org/10.1109/re.2008.37
12. Elattar H (2011) Intelligent information system to forecast the remaining life of aircraft turbofan engine. MSc. Thesis, Mansoura University, Mansoura, Egypt. http://www.eulc.edu.eg/eulc_v5/Libraries/Thesis/BrowseThesisPages.aspx?fn=PublicDrawThesis&BibID=379665
13. Goebel K, Saha B, Saxena A (2008) A comparison of three data driven techniques for prognostics. In: Proceedings of the 62nd meeting of the society for machinery failure prevention technology (MFPT), Virginia Beach, VA, USA, pp 119–131, May 6–8
14. Saxena A, Goebel K (2008) PHM08 challenge data set. NASA Ames prognostics data repository, NASA Ames Research Center, Moffett Field, CA. http://ti.arc.nasa.gov/project/prognostic-data-repository
15. Riad A, Elminir H, Elattar H (2010) Evaluation of neural networks in the subject of prognostics as compared to linear regression model. IJET-IJENS, 10(6):52:58. http://www.ijens.org/103706-5252%20IJET-IJENS.pdf
16. Heimes F (2008) Recurrent neural networks for remaining useful life estimation. In: 2008 international conference on prognostics and health management, October 6–9, Marriott Tech Center Denver, CO, USA. https://doi.org/10.1109/PHM.2008.4711422
17. Peel L (2008) Data driven prognostics using a Kalman filter ensemble of neural network models. In: 2008 international conference on prognostics and health management, October 6–9, MarriottTech Center Denver, CO, USA. https://doi.org/10.1109/PHM.2008.4711423
18. Wang T, Yu J, Siegel D, Lee J (2008) A similarity-based prognostics approach for remaining useful life estimation of engineered systems. In: 2008 international conference on prognostics and health management, October 6–9, Marriott Tech Center Denver, CO, USA. https://doi.org/10.1109/PHM.2008.4711421
19. Smith B (2011) Satellite enabled vehicle prognostic and diagnostic system. United States patent application publication US 2011/0046842 Al
20. Yan J, Koc M, Lee J (2004) A prognostic algorithm for machine performance assessment and its application. Prod Plan Control 15(8):796–801. https://doi.org/10.1080/09537280412331309208
21. Wang T, Lee J (2008) The operating regime approach for precision health prognosis. Failure Prevention for System Availability, 62th Meeting of the MFPT society, pp 87:98
22. Bonissone P (2006) Knowledge and time: a framework for soft computing applications in prognostics and health management (PHM), International symposium on evolving fuzzy systems, Ambleside, Cumbria, pp 19:24. https://doi.org/10.1109/ISEFS.2006.251159
23. Kalman R (1960) A new approach to linear filtering and prediction problems. Trans ASME J Basic Eng 82(1):35–45. https://doi.org/10.1115/1.3662552
24. Maybeck P (1979) Stochastic models, estimation and control, vol 1. Academic Press Inc, New York
25. Welch G, Bishop G (2006) An introduction to the Kalman filter, UNCChapel Hill, TR 95-041, July 24
26. Raspberry Pi (2015) https://www.raspberrypi.org/products/raspberrypi-2-model-b/