An Adaptive Framework For Remaining Useful Life Predictions Of Aircraft Systems

Marie Bieber[1], Wim J.C. Verhagen[2], and Bruno F. Santos[3]

[1,3] *Faculty of Aerospace Engineering, Delft University of Technology, Delft, 2629HS, The Netherlands*
M.T.Bieber@tudelft.nl
B.F.Santos@tudelft.nl

[2] *Aerospace Engineering and Aviation, RMIT University, Carlton, Victoria, 3053, Australia*
wim.verhagen@rmit.edu.au

ABSTRACT

Prognostics for condition-based maintenance does not only consist of prognostic algorithms but also involves steps such as data pre-processing, feature extraction, and dimensionality reduction, all of which contribute to the quality of the remaining useful life estimation. This process requires a lot of expertise and technical knowledge, which for many application systems is neither feasible nor affordable. In this paper, therefore, we present a generic framework with the capability to automatically choose the optimal settings for prognostics, given a specific data set. The framework consists of three phases. In the first one, a set of prognostic algorithms is selected by the user and according hyper parameter settings are found. In the second, a genetic algorithm optimizes the choice of methodologies together with hyper parameter settings for the feature extraction, dimensionality reduction, and prognostic algorithm selection. In the third phase, the identified settings define the prognostic setup, which in turn is used to train the model for remaining useful life estimation. This framework is then applied to a simulated aircraft engine data set. The aim is to find out if the framework provides the required adaptivity and furthermore if it allows to draw conclusions about the prognosability of the input data set. The first results show that the so obtained remaining useful life estimates are comparable to the values obtained using established prognostic algorithms on the same data set. Furthermore, the generic prognostic framework proves to be adaptive. In further consequence, such a generic framework offers a way to adjust a prognostic framework to different system data sets.

1. INTRODUCTION

Over the last few years the field of data-driven prognostics has undergone a big growth in terms of both algorithms and application areas. In many cases the development of a prognostic framework starts with a specific data set to which feature engineering methods and prognostic algorithms are tailored, ultimately resulting in increasingly better remaining useful life estimates. This process however not only requires a lot of expertise and technical knowledge, but also it often translates to years of research conducted. In case of an application system, such as an aircraft, which consists of many subsystems, this is neither feasible nor affordable. What would be desirable is a generic framework incorporating the steps needed for prognostics, including data pre-processing, feature engineering and the selection of a prognostic algorithm, that automatically chooses the optimal settings, given a specific data set.

Prior studies of such frameworks have yielded promising results. An autonomous diagnostics and prognostics framework is suggested by (Baruah, Chinnam, & Filev, 2006) that consists of several steps, including the data pre-processing, clustering to distinguish operating conditions and finally the diagnostics and prognostics steps are performed. However, some parameters, including the number of observations for initialisation and optimization of cluster adaption rates, have to be set manually and it can be tricky to tune the algorithm in an optimal way. To account for this, (Voisin, Levrat, Cocheteux, & Iung, 2010) provide a generic prognostic framework that can be instantiated to various applications. However, their approach is very formal and no specific machine learning algorithms are used in this framework. Again, this is a limitation, as it is up to the user to define proper techniques. To overcome this problem, (An, Kim, & Choi, 2015) provide guidelines to help with the selection of appropriate prognos-

tic algorithms depending on the application. Another way to address this is by using ensembles of machine learning approaches that combine multiple prognostic algorithms with an accuracy-based weighted-sum formulation (Hu, Youn, & Wang, 2010).

Still, a problem remains, namely the fact that the above presented solutions address the automatic selection of prognostic algorithms but not the steps needed before, such as feature extraction and dimensionality reduction. Therefore, the authors in (Trinh & Kwon, 2020) suggest a prognostics method based on an ensemble of Genetic algorithms that includes most of the steps, from feature engineering until the Remaining useful life (RUL) estimation. With this it provides a generic framework for prognostics. The authors of the paper validated their framework by applying it to three commonly used and available data sets and comparing its performance to other existing approaches. What we ask now is: Is the framework truly adaptive to various kinds of input data and does it adjust to those? And if so, can we go further and use the framework to make implications on the input data? Or put in other words: Does the framework provide the capability to impose criteria to the prognosability of a system? In this paper, we therefore apply a generic prognostic framework to a benchmark data set and test it in terms of adaptivity. From there we try to use the framework to loop back to the systems input data and make implications about the prognosability of those systems.

The remainder of this paper is organized as follows. Section 2 introduces the generic prognostic framework. In section 3, the for the case study selected underlying data set is described. Section 4 contains the results of the case study which are discussed in section 5. Finally, in section 6, we conclude by highlighting the most important findings and limitations and providing possible directions for further research.

2. GENERIC PROGNOSTIC FRAMEWORK

We define a generic prognostic framework to be a tool that contains modelling techniques covering the sequential steps of a data-driven prognostics approach and, given a data set, selects the best techniques to be used for each case. This means that in addition to incorporating different methodologies, the framework includes a selection step in which the in terms of prognostic error best set of techniques is chosen.

The idea of the generic prognostic framework as presented in this paper, is to provide for an approach capable of identifying a suiting prognostic model given related system data and a way to assess system data in terms of prognosability. This means that we do not make the assumption that according system data contains enough information to obtain a valuable prognostic model. It could indeed be the case that the generic prognostic framework fails to find a model of sufficient quality. This would mean that the system is not 'prognosable' given the available data set and the set of methodologies tested by the framework. Even though the 'prognosability' of the data set is therefore not a requirement for our framework, we still make some assumptions:

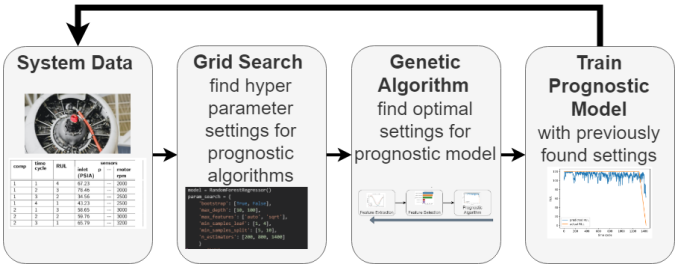

Figure 1. Generic prognostic framework (GPF)

- The system is operated until failure.
- According system data is available from the begin of operations until the failure.
- The remaining useful life (RUL) of the system is known at any time of operations.

There are multiple steps that have to be implemented in a prognostics framework, including the data-preprocessing, followed by feature extraction, leading to dimensionality reduction of features, before the actual prognostics algorithm that performs the remaining useful life prediction on the data set is executed. Therefore, a generic prognostic framework does not only need to provide the flexibility of choosing the 'best' prognostic algorithm, but also it has to incorporate the previous steps. Note, that we distinguish prognostic algorithms from prognostic models, in the way that when using the term 'prognostic algorithm' we refer to a certain selected technique used to do prognostics, e.g. Random forest (rf) or neural networks, and by 'prognostic model' we mean the derived predictor (by means of the prognostic algorithm and feature engineering methodologies) that takes system data as an input and outputs the RUL estimate.

Before we introduce the framework, we define the Mean squared error (MSE). At time T it is given as

$$MSE(T) = \frac{1}{T}\sum_{i=1}^{T}(RUL_i - \hat{RUL}_i)^2, \qquad (1)$$

with RUL_i the true RUL value and $\hat{RUL}_i$ the predicted RUL value at timestep i.

The objective is to select the in terms of MSE optimal methodology with the optimal hyper parameter settings for each step in the prognostic framework. We implement this in three basic steps summed up in Figure 1.

- First, a system is selected and according data is collected. Then, the hyper parameters for prognostic algorithms are selected by grid search.
- The next step and the heart of this study is solving the optimization problem that can be formulated as: Given a set

of feature extraction and dimensionality reduction techniques and their according hyper parameters, find the, in terms of MSE, optimal set of those given a prognostic algorithm. A detailed explanation of the set of feature extraction and dimensionality reduction methodologies is given in section 2.2.

- Finally, the settings are used to build the prognostic model to output the RUL estimate. The framework as suggested in this paper can be used in two ways: Either it can provide a quick assessment of the prognosability of the input data or it can be used to perform an automatic selection of feature engineering settings once the prognostic algorithm itself has been selected.

In the following subsections we present the grid search approach used to select the prognostic algorithms, in Section 2.1, the genetic algorithm used to optimize the feature engineering settings together with the prognostic algorithm, in Section 2.2, and the prognostic model that is the output of the framework in Section 2.3.

2.1. Grid search to tune prognostic algorithms

The first step in the proposed GPF is to select the prognostic algorithms. In this paper, we choose four different machine learning methodologies, a rf regression, a Multilayer perceptron (MLP), a Support vector machine (SVM) and a k-Nearest neighbors (kNN) regression. The four selected algorithms are well-established and offer potential advantages in terms of interpretability and explainability, which is necessary to understand systems retrospectively and prospectively (F. R. Ward & Habli, 2020). This may assist in the adoption of these algorithms for a variety of applications, potentially even covering safety-critical components. They thereby also provide the possibility to establish first baseline models for a quick prognostic assessment.

For the chosen algorithm on a validation set, a grid search is performed, to find the optimal hyper parameter settings. Since the aim of the grid search in this case is to establish quick baseline models that can consequently be used as in input in the following step of the framework, we only search a limited set of parameters. The according hyper parameters and their possible settings explored during the grid search are given in Table 1. The so found settings are the ones then used in the genetic algorithm that is presented in the next section.

2.2. Genetic Algorithm

We treat the problem of finding the optimal feature engineering settings as an optimization problem: The objective function is to minimize the MSE (Equation 1) of the fixed prognostic algorithm on the data set transformed by the selected feature engineering settings. To solve the optimization problem, we use a genetic algorithm (GA). These algorithms are based on the concepts of natural selection and genetics (Holland,

Table 1. The hyper parameters and combination of settings explored during the grid search for each of the four prognostic algorithms.

Prognostic algorithm	**Hyper parameter**	**Description**	**Possible settings**
rf	n estimators	number of trees	{200, 800, 1400}
	bootstrap	indicating if bootstrap samples are used to build the tree	{True, False}
	max depth	maximum depth of the tree	{10, 100, None}
	max features	maximum number of features to consider when looking for the best split	{'auto', 'sqrt'}
	min samples leaf	minimum number of samples required to be at a leaf node	{1, 2, 4}
	min samples split	minimum number of samples required to split an internal node	{2, 5, 10}
kNN	n neighbors	number of neighbors	{3, 5, 7, 9, 11, 19}
MLP	hidden layer sizes	number of neurons in the ith hidden layer	{(50, 50), (100, 50)}
	activation	activation function for hidden layer	{'tahnh', 'relu}
	alpha	learning rate	{0.0001, 0.001, 0.05}
	learning rate	schedule for weight updates	{'constant', 'adaptive'}
SVM	C	learning rate	{0.001, 0.01, 0.1, 10}
	gamma	kernel coefficient	{0.001, 0.01, 0.1, 1}

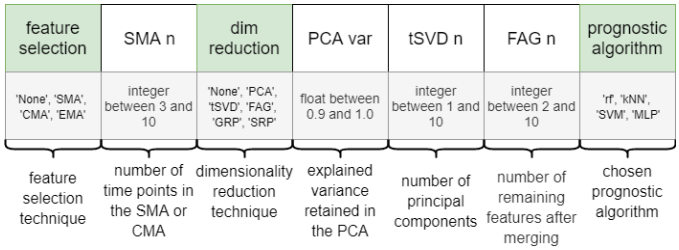

Figure 2. Example of chromosomes of an individual in the GA

1992). The steps, as in Algorithm 1, are as follows:

- A population is initialized, composed by a set of individuals (i.e., solutions to the optimization problem).
- The best fitted individuals are selected based on a fitness metric which represents the objective.
- In a following step, the selected individuals undergo a cross-over and mutation process to produce new children for a new generation of individuals.
- This process is repeated over a number of generations until the algorithm converges or a stopping criterion is achieved.

Algorithm 1: Genetic Algorithm

start;
$t \leftarrow 0$;
initialize population $P(t)$;
evaluate fitness of each individual in $P(t)$;
while *termination condition not fulfilled* **do**
  $t \leftarrow t + 1$;
  $s_1, s_2 \leftarrow$ select individuals from $P(t)$;
  $x_1, x_2 \leftarrow$ create offspring by crossover operation on s_1, s_2;
  $\hat{x_1}, \hat{x_2} \leftarrow$ mutate x_1, x_2;
  evaluate fitness of $\hat{x_1}, \hat{x_2}$ **if** *fitness of* $\hat{x_1}, \hat{x_2}$ *higher than least fittest individuals in* $P(t)$ **then**
    replace least fittest individuals with $\hat{x_1}, \hat{x_2}$;
  else
    pass;
  end
end

In our approach, we consider an individual as a possible set of methodologies and according hyper parameters of the prognostic framework. The fitness of each individual is given by the MSE at time T (Equation 1) resulting from the prognostics performed with the individuals settings on the underlying data set.

Each individual in the GA consists of a set of chromosomes, each representing a chosen methodology or hyper parameter setting. In the GA proposed, the individuals consist of seven chromosomes (Figure 2). Three of those correspond to choices of methodologies of the feature extraction and dimensionality reduction, and the others represent hyper parameters choices. With the settings as given in Figure 2, the solution space of the optimization problem corresponds to 691 200 possible solutions that are part of the solution space.

In the following subsections we give an overview of the multiple techniques considered by the GA for the feature extraction, feature section, and prognostic algorithm.

2.2.1. Feature extraction

Feature extraction is performed to obtain useful information from raw signals (Jardine, Lin, & Banjevic, 2006). Since the scope of this analysis are RUL estimation models for mechanical or electrical systems with run-to-failure data, it is assumed that underlying signals come in the form of time-series data. The simplest way to handle time series data is by calculating characteristic features as descriptive statistics from the data themselves. Of the existing methodologies, we chose to include three commonly used techniques for time series data in the framework, namely Simple moving average (SMA), Central moving average (CMA) and Exponential moving average (EMA). Assume that f_t is the value of feature f at time t and n is the considered time window size. Then the SMA is the average of values over the past points, given by

$$SMA(f_t) = \frac{f_{t-n+1} + \cdots + f_{t-1} + f_t}{n}. \quad (2)$$

The CMA is the average of the values around the time point t, i.e.

$$CMA(f_t) = \frac{f_{t-\frac{n}{2}} + \cdots + f_{t-1} + f_t + f_{t+1} \cdots + f_{t+\frac{n}{2}}}{n}. \quad (3)$$

And the EMA is recursively calculated by

$$EMA(f_t) = \begin{cases} f_t & \text{if } t = 1, \\ \frac{(1-\alpha)\cdot f_t + \alpha \cdot (1-\alpha^{t-1}) \cdot EMA(f_{t-1})}{1-\alpha^t} & \text{if} t > 1, \end{cases} \quad (4)$$

where $\alpha = e^{-1/n}$, with n the total number of time steps. The according hyper parameter settings in all three cases is the time window size n, which in the algorithm is represented by the variable `'SMA_n'`.

2.2.2. Feature selection

Feature selection is identifying features that help finding faults in the monitored systems (Kothamasu, Huang, & Verduin, 2009). Often this is achieved through dimensionality reduction techniques which generate a new lower-dimensional feature space while retaining information of the original features. In the framework, we included a sample of such techniques, namely Principal component analysis (PCA), truncated singular value decomposition (tSVD), Feature agglomaration (FAG), Gaussian random projection (GRP) and Sparse random projection (SRP). PCA is a widely used technique

making use of the singular value decomposition of the data to project it to a lower dimensional space. Truncated singular value decomposition is very similar to PCA it only differs in the way it treats the data matrix (Halko, Martinsson, , & Tropp, 2011). FAG uses Ward hierarchical clustering, grouping n_{FAG} features that are similar into clusters (J. H. Ward, 1963). Random projections are yet another way of reducing a features space dimensionality (Dasgupta, 2000). In GRPs this is achieved by means of a randomly generated matrix for which components are drawn from a Gaussian, with n the number of dimensions retained. SPRs differ from GRPs only by using a sparsely populated random matrix speeding up computations significantly (Li, 2007).

2.2.3. Prognostic algorithms

Finally, the according prognostic algorithm needs to be chosen and applied to the by the previous steps transformed data. The underlying set of algorithms with according hyper parameters consists of a rf regression, a MLP, a SVM and a kNN regression for which the hyper parameters were found during the grid search step as presented in section 2.1.

2.2.4. Genetic algorithm parameters

The previous paragraphs gave an overview over the form of an individual of the GA. Of course, also for the GA hyper parameters need to be set. The termination condition is chosen as the maximal number of generations. The probability with which an individual is mutated is set to 0.1, the probability for cross-over to 0.5 and the population size to 20 as presented in (Trinh & Kwon, 2020). With this, we are ready to run the GA and apply it to system data to find the 'optimal' settings of feature engineering methodologies and according hyper parameters. Now the next step is to use those settings to build the prognostic model.

2.3. Prognostic model

The output of the GA is the 'best individual', i.e. the set of methodologies and hyper parameter settings that lead to the best performance on the data set in terms of MSE. This individual is now used to build a prognostic model. As an input this model takes a new data set of according system data and it outputs the RUL estimation.

3. Case Study: Simulated aircraft engine data set

The data set we chose for this study is the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) data. It consists of four data sets, each containing simulated run-to-failure data for turbofan engines (Frederick, DeCastro, & Litt, 2007) (Saxena, Goebel, Simon, & Eklund, 2008). The data sets differ mainly in the number of fault modes ('modes') and operating conditions ('conditions') as listed in Table 2. Each engine is considered to be from a fleet of engines of the same type and each time series, also often referred to as trajectory, is from a single unit. The engines are operated until failure, i.e. the time series capture the operations of each unit until it fails. In the test set, the time series ends at some point before the failure and the objective is to estimate the RUL, or in other words the number of remaining operational cycles before failure. There are 21 sensor measurements and each row in the data contains the measurements corresponding to operations during one time cycle for a certain unit. Seven of those sensors are not used in the analysis, since they do not change over time and therefore do not show any trend.

Table 2. Characteristics of the four turbofan engine data sets, note that the difference between the four data sets lies within the number of fault modes ('modes') and operating conditions ('conditions')

Data set	#modes	#conditions	#Train units	#Test units
#1	1	1	100	100
#2	1	6	260	259
#3	2	1	100	100
#4	2	6	249	248

In order to train the prognostic models we require a labelled data set, i.e. we assume that the RUL is known at any time. In the C-MAPSS data set the units are operated until failure. Over the course of many studies performed on this data set it was found that instead of setting the RUL simply to the time to failure, a better performance is achieved when making the assumption that degradation will only start after the unit has been operated for a time. Therefore, the RUL is calculated using the piece-wise linear function that represents a constant RUL until 130 time cycles before failure, when the RUL is the linear function (i.e. time to failure) (Heimes, 2008).

4. Results

The aim of this study is to find out if the generic prognostic framework presented has the capability to adaptively adjust to different kinds of input data. Furthermore, we are interested in understanding the implications that can be made from the resulting prognostic models performance on the prognosability of the input data. In this section, we therefore perform two steps to answer the research questions. First, in Section 4.1 we establish a baseline and get a first insight into the GPF's dynamics by applying the framework to data set FD001. Second, in Section 4.2 a sensitivity analysis is performed on the input data to understand how adaptive the algorithm is and the implications of the results on the prognosability of the input data.

Table 3. Hyper parameter settings for the four prognostic algorithms on data set FD001

Prognostic algorithm	hyper parameter	Chosen setting
random forest	bootstrap	true
	min_samples_leaf	1
	min_samples_split	2
	n_estimators	100
k Nearest neighbors	leaf_size	30
	metric	'minkowski'
	n_neigbors	19
	p	2
	weights	'uniform'
Support vector machine	C	10
	epsilon	0.1
	kernel	'rbf'
Multilayer perceptron	activation	'relu'
	alpha	0.05
	hidden_layer_sizes	(50,50)

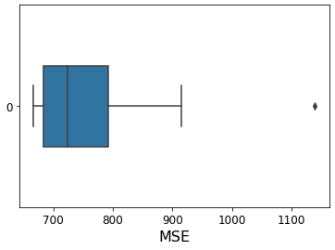

Figure 3. MSE (of 10 runs) for the prognostic frameworks found by the GA running for 50 generations on data set FD001.

4.1. Generic prognostic framework applied to C-MAPSS data set FD001

First, the GPF is applied to the C-MAPSS data set FD001. This is done for two reasons, namely to validate that the performance reached by the framework is indeed the same or higher than the performance reached of each of the single prognostic algorithms on each data set and to establish a baseline on this data set.

As a first step, a grid search is performed on the data set to find the hyper parameter settings for the four prognostic algorithms resulting in the settings listed in Table 3. Next, the GA is run for 50 generations and as indicated in section 2.2 with a population size of 20. Due to the instabilities of some of the techniques, the GA is run 10 times on the data set. This yields the prognostic settings given in Table 4. Those settings are used to build a prognostic model to estimate RUL. The according MSEs are also contained in Table 4 and shown in Figure 3.

When having a closer look at the selected methodologies in Table 4, note that the biggest differences lie in the chosen dimensionality reduction technique. The prognostic framework is in 9 out of 10 cases set to SVM and in the 10th case to MLP. The feature extraction technique is in most cases set to CMA, with different time window sizes. In three cases, SMA is the preferred choice. However, the trend towards a dimensionality reduction technique is not so clear: In three cases no dimensionality reduction is done, in three cases GRP is chosen, in two cases the preferred methodology is FAG and in the remaining cases it is set to SRP and PCA. The statistical properties of the according MSEs are visualized in Figure 3. They are mostly in a range of 650 to 800, with the exception of two outliers. The overall best performing prognostic framework settings of the ten runs performed on the original data set as presented in Table 4 are those found in run number 9. The feature extraction technique is set to CMA, with $n_{CMA} = 5$, FAG is selected for dimensionality reduction with $n_{FAG} = 2$ and SVM is the chosen prognostic algorithm. In further analysis, those are used as the benchmark settings on the data set FD001.

Table 4. Settings and MSE (of 10 runs) for the prognostic frameworks found by the GA running for 50 generations on data set FD001.

Run	MSE	feature extraction	Dim reduction	Prognostic algorithm
1	1140	None	GRP (n=4)	MLP
2	798	CMA (n =9)	FAG (n=4)	SVM
3	710	EMA (n=5)	GRP (n=7)	SVM
4	674	CMA (n=9)	GRP (n=7)	SVM
5	679	CMA (n=4)	SRP (n=9)	SVM
6	700	SMA (n=4)	None	SVM
7	776	SMA (n=9)	None	SVM
8	736	SMA (n=5)	None	SVM
9	666	CMA (n=5)	FAG (n=2)	SVM
10	916	CMA (n=6)	PCA (var=0.97)	SVM

Now, we use this established framework to compare the GPF to the models obtained using the four underlying algorithms. Also in this case, ten runs are performed to even out some of the algorithms' stochastic properties. The results are given in Table 5 and visualized in Figure 4. They indicate that indeed the SVM, which the prognostic framework chose as the preferred prognostic algorithm, is the best performing method

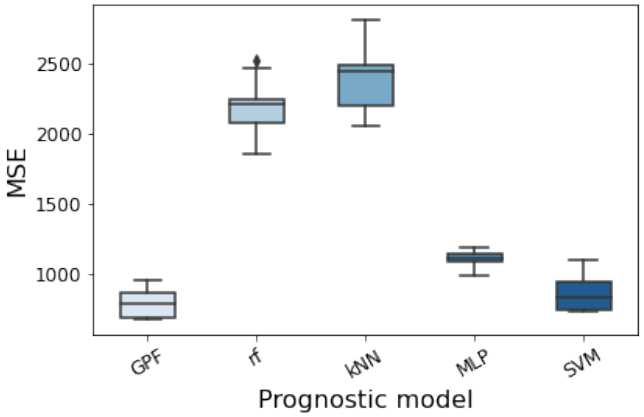

Figure 4. Comparison of the MSE (for ten runs) by the GPF obtained prognostic model to existing prognostic algorithms on data set FD001.

Table 5. Results (MSE on test and train set obtained after ten runs) of the by the GPF obtained prognostic model and existing prognostic algorithms on data set FD001.

Prognostic model	MSE (mean)	MSE (min)	MSE (max)
GPF	786	671	949
rf	2186	1850	2527
kNN	2378	2056	2819
MLP	1109	988	1186
SVM	860	725	1092

among the four prognostic algorithms used in this paper. It seems to be the case that the prognostic algorithm that achieves the best performance on the data set FD001 is the SVM. Therefore, for the following analysis, the prognsotic algorithm will be fixed to 'SVM'. This means that we slightly change the setup of the GA and fix the prognostic algorithm. This allows for a targeted sensitivity analysis on data characteristics and their influence on prognostics results.

4.2. Data sensitivity analysis

In this subsection, we performed a data sensitivity analysis to help us to answer the central research question addressed in this paper, namely if the framework can be used to assess input data in terms of prognosability. Additionally, the results can give us insight into how adaptive the framework is to different types of data sets. Essentially, we test this by addressing two questions, reflected in two tested scenarios:

1. How does the GPF adapt to having less data to train on?
2. How does the GPF behave if we use input data sets of different qualities?

Two corresponding scenarios are tested: To address Question 1 in Scenario 1 the data set size on which the prognostic models are trained is varied. Question 2 is addressed in Scenario 2, where we change the number of features used in the analysis.

Table 6. Settings and MSE (mean, min and max over ten runs) for the prognostic framework found by the GA running for 50 generations with different data set sizes.

# units	data set size	feature extraction	dim reduction	MSE (mean)	MSE (min)	MSE (max)
20	4168	EMA (n=6)	SRP (n=6)	1124.1	682	3283
40	7826	EMA (n=5)	SRP (n=4)	1181.2	678	3750
60	11942	SMA (n=7)	GRP (n=8)	1118.9	699	2856
80	16138	SMA (n=9)	GRP (n=4)	851.9	685	1904
100	20631	CMA (n=6)	FAG (n=2)	947.2	677	1985

4.2.1. Scenario 1: Varied training data set size

In this section we investigate how the framework adapts to various sizes of input data. Again, we perform this analysis on C-MAPSS dataset FD001, on which we already established a baseline in the previous section 4.1. As highlighted in section 3 the data set contains 100 trajectories in the train set, each representing data corresponding to the entire life of a unit. In the following, we refer to those trajectories as 'units'. In this scenario we vary the number of units contained in the train data set from 20 to 100 with steps of 20, resulting in five data sets of different sizes. The GA framework is implemented to run for 50 generations on each of those with a population size of 20. The resulting settings for the prognostic framework are given for each data set size in Table 6 as well as the MSE (for ten runs) of the so obtained prognostic models that can also be seen in Figure 5.

It is remarkable and becomes visible in Figure 5 that the settings found by the GPF for each of the data set sizes are very similar and so are the performances of the resulting models in terms of the mean MSE. Still, the worst performing models reach in case of using 20 units an MSE of 3283, while the MSE in worst case is 1985 when using all units in the train data set.

4.2.2. Scenario 2: Change in the number of features

To test the effect of input data of different qualitify, we run scenario 2. Since the C-MAPSS data set is widely used and many studies have already been performed on it, there exists quite a deep understanding of the underlying data and especially the features contained in it. As mentioned in sec-

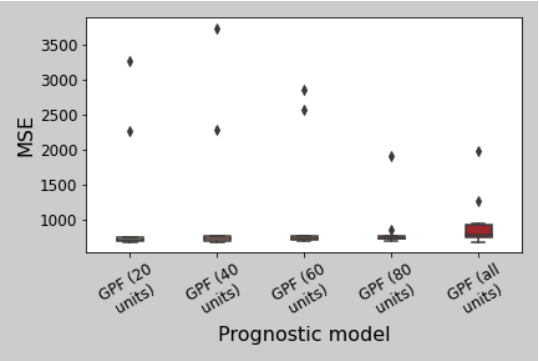

Figure 5. MSE (mean over ten runs) for the prognostic frameworks found by the GA running for 50 generations for different train data set sizes.

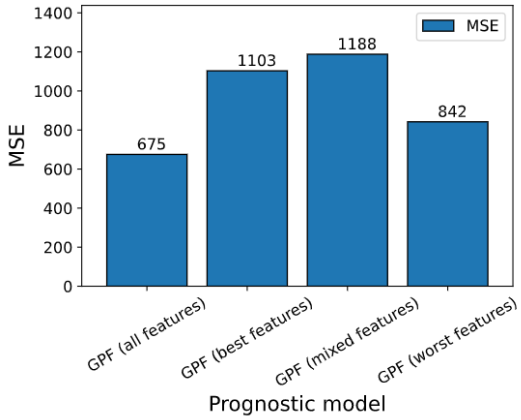

Figure 6. MSE for the prognostic framework found by the GA running for 50 generations for different qualities of input data.

tion 3, there are 21 features, corresponding to sensor readings and 7 of those are not considered in this study because they are constant, leaving us with 14 features corresponding to sensors $2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, 21$. It has been found that of those seven sensors, namely sensors $7, 8, 9, 12, 16, 17$ and 20 are the most valuable ones for RUL estimations (Wang, Yu, Siegel, & Lee, 2008). Based on their findings, we choose three sets of features as follows:

- a In setup a, we the subset of most relevant features, i.e. including sensors $7, 8, 9, 12, 16, 17$ and 20.
- b In setup b, we perform the analysis using a subset of features, including a few of the relevant sensors and a few of those with low to no prognostic value, i.e. sensors $7, 8, 9, 2, 3$ and 13.
- c Setup c consists of a subset of sensors of little to no predictive value, namely sensors $2, 3, 11, 13, 14$ and 15.

In all setups the GA is run for 50 generations, with a population size of 20. Again, we compare the results of ten runs to captures instabilities in techniques. The settings found by the GPF for the three sets of features are given in Table 7. The

Table 7. Settings for the prognostic framework found by the GA running for 50 generations with different data set qualities.

setup	**features in data set**	**feature extraction**	**dim reduction**	**MSE**
a	7, 8, 9, 12, 16, 17, 20	None	FAG (n=3)	1103
b	7, 8, 9, 2, 3, 13	SMA (n=6)	PCA (var = 0.93)	1188
c	2, 3, 11, 13, 14, 15	CMA (n=6)	None	842
baseline	all	CMA (n=6)	FAG (n=2)	675

MSE found by the prognostic models build using the given settings can also be found in the table as well as in Figure 6. The baseline scenario over all has the best performance. However, it is followed by setup c, in which no dimensionality reduction technique is used and which is the setup with the lowest input data quality. Setup a performs slightly better than setup b, but overall all the thereby found prognostic models perform in a similar range in terms of MSE.

To get more insight into the dynamics behind the results, we have a closer look at the features selected during the dimensionality reduction. For the benchmark GPF the chosen dimensionality reduction technique is a FAG with two clusters (Table 4). The features contained in the first cluster are sensors $2, 3, 4, 7, 8, 9, 11, 12, 13, 14, 15, 17, 20, 21$ and the feature contained in the second cluster is sensor 16. In setup a, i.e. when running the GA framework on the best set of features, the chosen dimensionality reduction technique again turns out to be FAG with the number of clusters set to $n_{FAG} = 3$. In this case, the features in the first agglomeration are sensor 9 and 16, the ones in the second cluster are sensors $7, 12$ and 20 and sensors 8 and 17 are in the third cluster. In case of using a set of mixed features (setup b), PCA with the variance retained set to 0.93 is used as dimensionality reduction technique. The features selected in this case are sensor 9 and 7. For setup c when using a subset of sensors of little to no predictive value, the GA framework finds that the best performance is reached when no dimensionality reduction is performed at all.

5. Discussion

The aim of this paper, is two fold: First, it provides an assessment of the presented generic prognostic framework in terms of adaptivity. Second, we intend to find out how the framework can be used to identify if input data sets have the potential to be used for prognostics or if the size of the training data set e.g. is too limited to perform prognostics using those

data sets. In Section 5.1, we discuss the results obtained in Section 4 and how they contribute to answering our research questions. All the questions that remain or arise from the results are addressed in Section 5.2 as are the limitations of our study and opportunities for further research.

5.1. Discussion of the results of the case study

In Section 4.1, by applying the GPF to the C-MAPSS data set FD001, some dynamics of the framework became clear. Some of those are also important for the more interesting results of section 4.2, therefore we will address them first.

In light of the obtained results, the most important property of the GPF to consider is its stability (Table 4). While over the ten runs the GPF gave quite consistent choices in terms of feature extraction and prognostic algorithm methodology, the same is not true for the chosen dimensionality reduction techniques. For the MSE as it becomes clear when looking at Figure 3, the results are mostly stable in a range of 650 to 800. Only in two runs the performance of the GPF dropped, namely run 1 and run 10 (Table 4). In run 1 this can clearly be traced back to selecting MLP as the chosen methodology, especially when comparing the results with those that can be observed in Figure 4. The average performance of prognostic models build using MLPs is an MSE of 1109, while it is 860 for the SVM. In run 10, it seems as if the choice of using PCA as a dimensionality reduction technique is the reason for the low performance. When used together with FAG, the CMA for the feature extraction and SVM reach the highest performance.

Another important finding appears in Table 5 and are visualized in Figure 4. We can, in fact, learn two lessons from those: First, the GPF outperforms the existing prognostic algorithms when used stand alone, which is a sort of validation, because this lies in the nature of how the GPF was built. The second lesson, however, is more interesting: The different techniques reach very different MSEs on the data sets. They range from a mean MSE of 860 for SVMs to an MSE of 2378 of the kNN. This is rather unexpected and is not really in alignment with what was found previously on the data set (Zhang, Lim, Qin, & Tan, 2017). The reason for this could be the set of hyper parameters explored during the grid search.

Next, we explore the results found during the sensitivity analysis in section 4.2 and look into what implications can be made for the adaptivity of the framework and the loop back to the input data and their prognosability. For this purpose, we first address scenario 1, i.e. changing the train data sizes (section 4.2.1) and then scenario 2, i.e. the effect of changing the quality of the input data set (section 4.2.2). As shown in Table 6 and Figure 5 there is no significant difference in the outputted prognostic model's performance. They only differ in the worst performance reached by a model over ten runs, which decreases from 3283 for 20 units, to 2856 for 60 units to 1985 for 100 units. This is what one would expect also for the mean of the MSE, but instead it hovers around 1100 for 20, 40 and 60 units and only drops to around 850 for 80 units and 950 when using all units. The answer to this could be linked to the answer of our research question: By dynamically choosing feature engineering methodologies, the framework adapts to smaller train data sets. Indeed, for the two smallest data sets the feature extraction technique was always set to EMA and the dimensionality reduction technique to SRP. Opposed to that for the two bigger data set sizes, the preferred features selection technique was SMA and the dimensionality reduction technique GRP.

Equally interesting are the results found for scenario 2 given in Table 7 and Figure 6. Here, the by the GPF found prognostic models performance ranges from 675 for the baseline GPF that uses all features to 1188 for the GPF that was trained on a constrained set of features containing for prognostics relevant and irrelevant features. The prognostic model found on the set of relevant features performed with an MSE of 1103 only slightly better and the prognostic model found on a set of as irrelevant regarded features with an MSE of 842 performs best. We found three possible explanations for this behaviour: The first makes use of a finding from section 4.1, namely the instability of the framework in terms of dimensionality reduction technique chosen. In Table 7 it can be seen that the three methodologies chosen for feature extraction for those setups are all different from each other, leading to very different performances overall. However, when we look deeper into the features that were chosen by the respective dimensionality reduction techniques, those features are all in alignment with the findings by (Wang et al., 2008). For setup b, e.g. the PCA selects sensors 9 and 7, two of the as relevant regarded features. Another explanation for this behaviour could simple be the fact that we ran the GA only one time using those settings and as we could observe in Table 4, the MSE between different runs can change by margins of around 500. Yet another explanation could be the fact that the features of sensors $2, 3, 11, 13, 14$ and 15 contain important prognostic information after all. This is highly unlikely, however, since they at least do not contain any visible trends (Wang et al., 2008).

With all of those findings, we can assess the second question we asked in the beginning of section 4.2, which is also related to our central research question. Scenario 2, very like scenario 1 showed that the framework indeed adjusts to input data of various sizes and qualities. Due to its adaptivity and due to instabilities mainly in the choice of the dimensionality reduction technique, it is almost impossible to make implications about the input data from the results of the GPF. The only way to do so is by assessing the choices it makes in terms of feature extraction and dimensionality reduction. However, further research is recommended to address these issues.

5.2. Limitations and further research

The GPF as presented in this paper, intends to give a first overview over the possibilities that lie within applying such a framework to various system data and enable a quick prognostic assessment on this data set. We do not claim that it is exhaustive or contains all steps and methodologies that should be included in a very generic framework. This is certainly a limitation. There are various more steps in prognostics that can have quite an impact on the quality of RUL estimations, such as enhanced health estimations or diagnostics (Elattar, Elminir, & Riad, 2016).

Furthermore, although the GPF can give an indication of the prognosability of a system, it should be kept in mind that this is only an indication. In case a system, based on the underlying data, turns out to be not prognosable, it could still be the case that a change in system data, like adding more sensors or measuring in different time intervals, provides prognosable data for this system. It is important to note that while the framework provides the capability to assess prognosability, it does so based only on the available system data.

Similarly, the representative prognostic algorithms chosen in this study are only a few selected algorithms of many. We assume that prognostic algorithms of sufficient quality exist. The idea of the framework, however, is that whatever prognostic algorithm is used as an input, it would be able to find the respective optimal feature engineering settings. A future research could be to test the framework on a set of more recent prognostic algorithms that were shown to perform quite well on this data set, such as LSTM-CNNs (Jayasinghe et al., 2018). Finally, to explore the adaptivity and advantages of the GPF further, it would be interesting to extend the analysis to real system data.

6. Conclusion

We applied a generic prognostic framework to a publicly available prognostic benchmark data set to assess its flexibility to deal with changes in data sets, understand how it adapts to those changes and draw conclusions towards the prognosability of systems. Two important findings were observed: First, the framework is highly adaptive to changes in both, the data quality and size. Second, drawing conclusions about input data and a systems prognosability is tricky but not impossible. One thing is clear, the prognostic models performance alone does not give enough clues about how prognosable a system is. However, when looking deeper, into the settings the algorithm comes up with, then we could observe certain trends, especially with regards to data set sizes. Furthermore, the framework has the potential to be used to find the optimal feature engineering settings, given a prognostic algorithm. To fully explore this and understand the GPFs dynamics even in more detail, we suggest to further extend the research to real system data.

Abbreviations

C-MAPSS Commercial Modular Aero-Propulsion System Simulation. 5, 6, 7, 9, 10

CMA Central moving average. 4, 6, 9, 10

EMA Exponential moving average. 4, 9, 10

FAG Feature agglomaration. 4, 5, 6, 8, 9, 10

GA Genetic algorithm. 10

GPF Generic prognostic framework. 2, 3, 5, 6, 7, 8, 9, 10

GRP Gaussian random projection. 4, 5, 6, 9, 10

kNN k-Nearest neighbors. 3, 5, 9, 10

MLP Multilayer perceptron. 3, 5, 6, 9, 10

MSE Mean squared error. 2, 3, 4, 5, 6, 7, 8, 9, 10

PCA Principal component analysis. 4, 5, 6, 8, 9, 10

rf Random forest. 2, 3, 5, 10

RUL Remaining useful life. 2, 3, 4, 5, 6, 8, 10

SMA Simple moving average. 4, 6, 9, 10

SRP Sparse random projection. 4, 6, 9, 10

SVM Support vector machine. 3, 5, 6, 7, 9, 10

tSVD truncated singular value decomposition. 4, 10

Acknowledgment

This research is supported by European Union's Horizon 2020 program under the ReMAP project, grant No 769288.

References

An, D., Kim, N. H., & Choi, J. H. (2015). Practical options for selecting data-driven or physics-based prognostics algorithms with reviews. *Reliability Engineering and System Safety*, *133*, 223–236. Retrieved from http://dx.doi.org/10.1016/j.ress.2014.09.014 doi: 10.1016/j.ress.2014.09.014

Baruah, P., Chinnam, R. B., & Filev, D. (2006). An autonomous diagnostics and prognostics framework for condition-based maintenance. *IEEE International Conference on Neural Networks - Conference Proceedings*, 3428–3435. doi: 10.1109/ijcnn.2006.247346

Dasgupta, S. (2000). Experiments with Random Projection. *Proceedings of the Sixteenth conference on Uncertainty in artificial intelligence (UAI'00)*, 143–151. Retrieved from http://arxiv.org/abs/1301.3849

Elattar, H. M., Elminir, H. K., & Riad, A. M. (2016). Prognostics: a literature review. *Complex & Intelligent Systems*, *2*(2), 125–154.

Frederick, D. K., DeCastro, J. A., & Litt, J. S. (2007). User's guide for the commercial modular aero-propulsion system simulation (C-MAPSS).

Halko, N., Martinsson, P.-G., , & Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM review*, *53*(2), 217–288.

Heimes, F. O. (2008). Recurrent neural networks for remaining useful life estimation. *2008 International Conference on Prognostics and Health Management, PHM 2008*(November 2008). doi: 10.1109/PHM.2008.4711422

Holland, J. H. (1992). *Adaptation in Natural and Artificial Systems*. doi: 10.1016/S0376-7361(07)53015-3

Hu, C., Youn, B. D., & Wang, P. (2010). Ensemble of data-driven prognostic algorithms with weight optimization and k-fold cross validation. *Proceedings of the ASME Design Engineering Technical Conference*, *3*(PARTS A AND B), 1023–1032. doi: 10.1115/DETC2010-29182

Jardine, A. K., Lin, D., & Banjevic, D. (2006). A review on machinery diagnostics and prognostics implementing condition-based maintenance. *Mechanical Systems and Signal Processing*, *20*(7), 1483–1510. doi: 10.1016/j.ymssp.2005.09.012

Jayasinghe, L., Samarasinghe, T., Yuen, C., Chen, J., Low, N., & Ge, S. S. (2018). Temporal Convolutional Memory Networks for Remaining Useful Life Estimation of Industrial Machinery. *arXiv preprint arXiv:1810.05644*.

Kothamasu, R., Huang, S. H., & Verduin, W. H. (2009). System health monitoring and prognostics - A review of current paradigms and practices. *Handbook of Maintenance Management and Engineering*(July 2006), 337–362. doi: 10.1007/978-1-84882-472-0_14

Li, P. (2007). Very sparse stable random projections for dimension reduction in lα (0 ¡α 2) norm. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 440–449. doi: 10.1145/1281192.1281241

Saxena, A., Goebel, K., Simon, D., & Eklund, N. (2008). Damage Propagation Modeling for Aircraft Engine Prognostics.

Trinh, H. C., & Kwon, Y. K. (2020). A data-independent genetic algorithm framework for fault-type classification and remaining useful life prediction. *Applied Sciences (Switzerland)*, *10*(1). doi: 10.3390/app10010368

Voisin, A., Levrat, E., Cocheteux, P., & Iung, B. (2010). Generic prognosis model for proactive maintenance decision support: Application to pre-industrial e-maintenance test bed. *Journal of Intelligent Manufacturing*, *21*(2), 177–193. doi: 10.1007/s10845-008-0196-z

Wang, T., Yu, J., Siegel, D., & Lee, J. (2008). A similarity-based prognostics approach for remaining useful life estimation of engineered systems. *2008 International Conference on Prognostics and Health Management, PHM 2008*(November). doi: 10.1109/PHM.2008.4711421

Ward, F. R., & Habli, I. (2020). An Assurance Case Pattern for the Interpretability of Machine Learning in Safety-Critical Systems. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *12235 LNCS*, 395–407. doi: 10.1007/978-3-030-55583-2_30

Ward, J. H. (1963). Hierarchical Grouping to optimize an objective function. *Journal of the American Statistical Association*, *58*(301), 236–244.

Zhang, C., Lim, P., Qin, A. K., & Tan, K. C. (2017). Multiobjective Deep Belief Networks Ensemble for Remaining Useful Life Estimation in Prognostics. *IEEE Transactions on Neural Networks and Learning Systems*, *28*(10), 2306–2318. doi: 10.1109/TNNLS.2016.2582798

BIOGRAPHIES

Marie Bieber currently pursues her PhD at Delft University of Technology, Netherlands. She gained her Master in Mathematics from the Vienna University of Technology, Austria. Her research interests are prognostics in the framework of Condition-Based Maintenance and the evaluation of developed CBM solutions for different stakeholders.

Wim J.C. Verhagen is a Senior Lecturer within the Aerospace Engineering and Aviation Discipline in the School of Engineering, RMIT University. Dr Verhagen's research covers predictive maintenance and decision support. With respect to predictive maintenance, Dr Verhagen's research efforts focus on the development, implementation and validation of data-driven and hybrid diagnostic and prognostic models for aircraft systems. With respect to decision support, his research covers a comprehensive and integrated set of models to streamline maintenance planning, execution and control on operational and tactical time horizons.

Bruno F. Santos is an assistant professor at the Delft University of Technology, Faculty of Aerospace Engineering, in the group of Air Transport and Operations. His research focuses on the development of decision-support tools to support adaptive airline operations, combining optimization methods with machine learning techniques. His work entails the development of optimization frameworks to improve aircraft availability by optimizing aircraft maintenance schedules, plan airline network and fleet development, schedule aircraft and crew operations, and manage disruptions during operations.