Prediction of Jet Engine Parameters for Control Design using Genetic Programming

Giovanna Martínez-Arellano and Richard Cant
School of Science and Technology
Nottingham Trent University
Nottingham, U.K.
Email: N0204624@ntu.ac.uk

Lars Nolle
Department of Engineering Sciences
Jade Hochschule/University of Applied Science
Wilhelmshaven, Germany
Email: lars.nolle@jade-hs.de

***Abstract*—The simulation of a jet engine behavior is widely used in many different aspects of the engine development and maintenance. Achieving high quality jet engine control systems requires the iterative use of these simulations to virtually test the performance of the engine avoiding any possible damage on the real engine. Jet engine simulations involve the use of mathematical models which are complex and may not always be available. This paper introduces an approach based on Genetic Programming (GP) to model different parameters of a small engine for control design such as the Exhaust Gas Temperature (EGT). The GP approach has no knowledge of the characteristics of the engine. Instead, the model is found by the evolution of models based on past measurements of parameters such as the pump voltage. Once the model is obtained, it is used to predict the behaviour of the jet engine one step ahead. The proposed approach is successfully applied for the simulation of a Behotec j66 jet engine and the results are presented.**

***Keywords*-genetic programming; jet engine; simulation;**

I. Introduction

The performance of a jet engine depends in most part on the design of its control system [1]. The mathematical modelling of a jet engine is significant for the controller plan. Designers need to have a mathematical model to investigate efficient control algorithms. Control systems may be conservative if the designer does not have a complete model of the engine, forcing the system to regulate measurable variables and operating the engine in a less efficient manner. In early control designs, designers used experimental methods to find the steady state or transient behaviour of an engine. This method was time consuming and expensive. Therefore, the first attempts to model an engine behavior using mathematical and thermodynamic equations began to appear [2][3][4]. Mathematical models were developed by using physical rules and empirical data. Thermodynamic properties of combustion gases and air were calculated using variation of the temperature. One of the earliest published work that attempted to quantify engine dynamics was that of the National Advisory Committee for Aeronautics (NACA) where Gold and Rosenzweig showed that the rotor of a turbojet responded to sudden changes in fuel flow as a first-order system [5]. Lawrence et al whose analysis assumed that a sudden increase in fuel flow could cause an instantaneous increase in turbine torque but zero increase in compressor torque was a major advance [6]. Then the first simulations began. Larrowe and Spencer did a first attempt of using a model operating in real time to develop control hardware using a simulation rather than an actual engine [7]. As simulation became an important tool on the design of control systems, some programs such as Matlab/Simulink became very popular for modelling and simulating dynamic systems [8][9].

However, as the level of sophistication of modern jet engines increased, the complexity of the mathematical models required for an accurate simulation grew. This led to the application of many knowledge-based systems such as Fuzzy Logic (FL), neural networks (NN), genetic algorithms (GA) and probabilistic reasoning (PR) for the generation of complex models. Modern control techniques for full-sized jet engines may contain some of these $soft\ computing$ techniques to improve the propulsion system performance [10]. Some of these techniques have also been applied for the development of diagnostic and prognostic systems that can predict the progression of various faults to component failure [11][12]. Nayyeri et al. propose an offline health monitoring system by simulating the EGT using GP [13]. EGT is an engine operating limit that can be used to monitor overall engine operation conditions.

In recent years small scale jet engines, which operate on the same principles as the commercial jet engines have been developed. Initially this work was done by amateurs for use in model aircraft [14] but more recently commercially produced engines have been produced and used for research and education purposes. Developing robust control systems is not practical without a simulation, especially if soft computing techniques are used. Testing preliminary versions of an evolutionary controller may be cause of potential damage to the engine due to their randomness. This is in addition to the impracticality of extended periods of test running without specialist facilities. For an amateur designer, developing a good controller without complete knowledge of the mathematical model would be a challenge. However, following a design/simulation/data acquisition iterative approach, could facilitate the process. This paper proposes the use of genetic programming as a mean of constructing a simulation of the engine for control design. The rest of the paper is organised as follows: Section 2 presents genetic programming as a symbolic regression tool and introduces

978-1-4799-4923-6/14 $31.00 © 2014 IEEE
DOI 10.1109/UKSim.2014.64

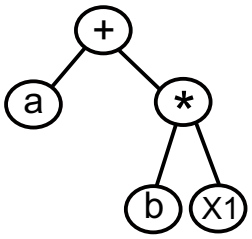

Figure 1. Equation represented by the tree: $a + bx_1$

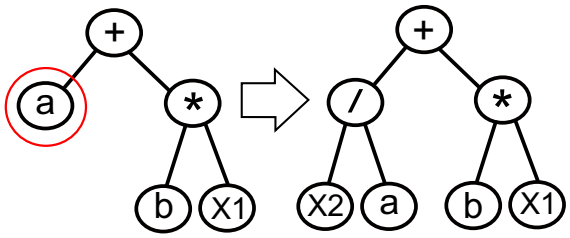

Figure 2. Mutation Operation

the simulation approach using GP to estimate different engine parameters. Section 3 presents the experiments and discussion and finally Section 4 presents the conclusion and future work.

II. Genetic Programming for Symbolic Regression

Genetic Programming [15] is a biologically inspired computation technique based on the evolution of individuals over time, through events such as crossover and mutation, which progressively refines them into better individuals. In GP, a population of programs (in a binary tree layout) is evolved, each program representing a set of instructions to solve a specific problem. GP, like nature, is a stochastic process, which cannot guarantee to find the global optimum but it is that randomness which can lead it to escape local optima, which deterministic methods may be captured by [16]. Symbolic regression via GP is a non-parametric, non-linear regression technique that looks for an appropriate model structure and model parameters as opposed to classic regression that assumes a certain model structure and estimates the optimal parameters that best fit a given sample of data [17]. As shown by the example in Figure 1, a GP tree is formed by a set of terminals and functions. The functions may be basic arithmetic operators {+, -, *, /}, standard mathematical functions (sine, cosine, logarithmic, exponential), logical functions or domain-specific functions. The terminals may be constants or any problem-related variables. For the jet engine simulations, variables such as pump voltage, pressure, exhaust gas temperature, among others, may be relevant for behavioural modelling. The evolution process will be able to identify those variables that are relevant for the model.

The GP algorithm is implemented as follows: First, an initial population p of n trees is created, selecting random combinations of variables and operators. There are different methods for creating random trees. A very common method is the grow method, which creates trees of different depths up to a specified maximum. Another method is the full method which creates trees were the terminals are guaranteed to be at a certain depth. A third method, which was used in this paper, is the ramped half-and-half which uses both grow and full methods to generate a population with good variation. The fitness of each tree is calculated based on the RMSE between the output given by the model on each training point and the real observation (Equation 1) plus the penalisation function (Equation 2). Once the tree with the best fitness is identified, the algorithm iterates x number of generations. At each iteration, a new population is created by copying the best tree of the previous generation and the rest of the individuals are created by means of selection, crossover and mutation operators. The new population replaces the previous population and the best tree of the new population is identified. This process is repeated until the specified number of generations has been reached and the best tree on the training set and validation sets are obtained.

$$RMSE = \sqrt{\sum_{i=0}^{s} (observation - prediction)^2} \quad (1)$$

A. Mutation Operator

This genetic operator consists in replacing a randomly selected subtree from a complete tree with a new randomly generated subtree. Figure 2 shows an example of this operator. Every new individual in a population is a candidate for mutation, however this depends on a certain probability. The mutation probability is a parameter that needs to be tuned during experimentation and it is usually set to very small values.

B. Crossover Operator

The crossover operator consists of copying the material from two selected trees to generate two new trees. First each parent is selected via a tournament. The tournament consists of selecting a subset (i.e. 20) of trees and the tree with the best fitness wins the tournament. Two tournaments are needed to selected a pair of parents. Once these are obtained, a subtree is selected randomly from both trees. These two subtrees are exchanged generating two new trees. Figure 3 shows an example of this operator.

C. Overfitting

As any other Machine Learning (ML) technique, achieving good generalization is one of the most important goals of the GP approach. Failure to generalize, or overfitting, happens when the solution performs well on the training cases

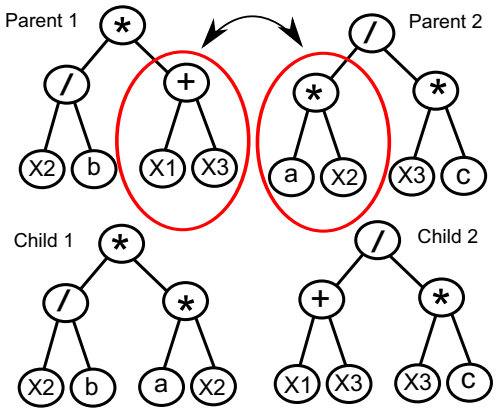

Figure 3. Crossover operation

Figure 4. Pressure prediction one time step ahead with the best model found using pump voltage on test data

but poorly on the test cases. The most common approaches to reduce overfitting consist in biasing the search towards shorter solutions. The parsimony pressure technique [17], penalises the fitness of a program according to its complexity (number of nodes in the tree), reducing the probability for it to be selected in future generations for crossover. The mathematical representation of the penalisation function used is shown on Equation 2. The first term of the fitness equation is the sum of the errors between the obtained output (new forecast) and the desired output (observations) in the training set (training set size = s). The second term is the complexity factor, where t is the number of nodes of the GP tested and k is a trade-off weight that allows to control the level of pressure of the complexity factor. A single value of k do not work for all regression cases. Depending on the variable to model, is the range of values of k. In general, high values of the parsimony pressure lead to less complex models and low values of the parsimony pressure lead to larger, more complex models.

$$f = \frac{1}{s}\sum_{i=0}^{s} e(i) + k\left(\frac{(t^2 log_2(t)}{s}\right)^{\frac{1}{2}} \quad (2)$$

D. GP for the prediction of a jet engine parameters

As mentioned previously, GP has been widely applied for symbolic regression. In a previous work, GP was successfully applied as a downscaling technique for wind speed forecasting [18]. For this reason it is interesting to explore the application of this approach for modelling different jet engine parameters without knowledge of the complex physics of it. The approach presented is as follows: first, with a simple fuel flow controller, pump voltage, pressure, temperature among other data is gathered from the engine. The GP approach is then used to generate a case pressure model using pump voltage and pressure observations. A second model is generated applying the GP approach to model the rotor speed from the case pressure and the pump voltage. In a following stage, with EGT observations a third model is developed using the pump voltage, the pressure and the rotor speed to predict future values of EGT. Once these models are obtained, they can be used to simulate the behaviour of these parameters when testing a new control design. The first model would predict the case pressure from the pump voltage. Then, with the predicted pressure, a prediction of the rotor speed can be obtained and consequently a prediction of the EGT can be obtained as well. The predictions are only one step ahead, so as soon as these are obtained, they can be fed back into the model to obtain the next prediction.

III. Experimental Setup

The starting point for the experiments was a set of data that had been logged during previous runs of the Behotec j66 engine (Figure 4). The information available consisted of pump voltage, case pressure rpm and Exhaust Gas Temperature (EGT) logged at intervals of 0.2 seconds. The experimental setup was divided in three stages. The first stage involved the use of pump voltage observations in order to predict the case pressure. To keep the number of inputs as small as possible, four input values of the pump voltage were used. The first one was the actual observation at time t. The second input was the average of the four last observations of the pump voltage ($t = t, t-1, t-2, t-3$). The third input was the average of the last 8 observations ($t = t, t-1, ..., t-7$) and the fourth input was the average of the last 16 inputs. In this way, a general overview of the changes in the variable from the last 18 time steps could be considered without increasing to 18 the number of inputs to the algorithm. The expected output would be the pressure at time t. Previous values of the pressure were not used to predict pressure at time t. The reason for this was that very small differences of the variable in different time steps could lead the search of the algorithm to pick up faster these previous values of the pressure rather than other variables because the algorithm is trying to decrease the RMSE between the forecast and the observation.

The second stage used the pressure and pump voltage in order to predict the rotor speed. For the second set of experiments, the number of inputs increased to 8 in order to consider the four values for both the pump voltage and the pressure. As it was done with the previous experiments, the rpm parameter was not fed into the algorithm to avoid picking up this variable over the other ones.

Finally, for the third set of experiments two settings were used. In a first attempt, pump voltage, pressure and rpm were used. For each variable, the four inputs as used in previous experiments were used, having a total of 12 inputs. In a second setting, history of the exhaust gas temperature was included. Due to the low correlation between the temperature and these three variables, it was also necessary to include historical data from the temperature. Three averages of the temperature were considered. An average on the previous 32 readings, an average on the last 16, and an average on the last 8. A longer history period was used for the temperature as this variable is changing at a lower pace compared to the other three variables. The total number of inputs to the algorithm grew to 15. Genetic programming is very sensitive to the input data. Feeding all variables to all experiments is not always the best approach. So it must be decided intuitively which variables are most important and the input must be limited to these ones only.

For the three sets of experiments the training and testing sets were designed as follows. Several runs of the engine were used. Those variable readings obtained by the sensors where the engine and/or pump were not running were excluded from the recorded information because they were unrepresentative. The remaining data was divided into training and testing sets. Both training and testing files contained data from startup and operation stages. From the total amount of records available 60% were used for training and 40% for testing. The training set was further randomly divided into 80% training and 20% validation.

IV. Experimental Results and Discussion

Once the training, validation and test sets were obtained, the experiments were carried out as follows. First, for each type of experiment the best values of k were identified empirically. This parameter allows to control the complexity of the models by adding a penalisation to the fitness based on their size. Each modelled variable has its own complexity so different values of k were applied. For the case pressure variable, a parsimony pressure of $k = 0.005$ was enough to avoid very complex trees. The raw fitness (fitness with no penalisation) of the models obtained for the case pressure is 0.045 on average, so the parsimony pressure is such that does not provide an unwanted advantage over raw fitness. Larger models will be allowed to survive only if they provide a significant improvement over the raw fitness even when adding the penalisation. To model the rotor speed, the parsimony pressure used was around $k = 80$. For the EGT, values between $k = 0.2$ and $k = 0.3$ were used. Once these parsimony pressures were obtained they were used to execute 50 runs of each type of experiment. At each run, the training and validation sets were selected randomly. The additional parameters that were set for the experiments are shown in Table I which are typical settings for the GP approach. The best models found for each variable are shown in Table II. The subindexes of the variables are used to denote the type of input. A subindex of 1 means the current value of the variable at time t, 2 denotes the average of the last 4 observations, 3 corresponds to the last 8 observations and 4 to the last 16 observations. The models were applied to the test sets to evaluate the quality of the predictions.

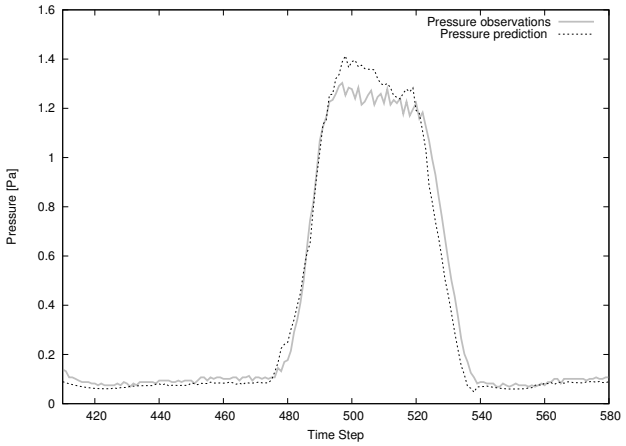

Figure 5. Pressure prediction one time step ahead with the best model found using pump voltage on test data

Figure 5 presents the predictions of the case pressure obtained on the testing set with the best model found on the first set of experiments. The correlation between the observations and the predictions is very high, so the algorithm easily detected the relationship among them. Figure 6 presents the predictions of the rpm parameter using the best model found on the second set of experiments. It can be seen that these relationship was also quite well caught by the algorithm producing an accurate model.

Figures 7 and 8 present the predictions of the exhaust gas temperature using the best model from the third set of experiments. The trend of the temperature is well caught in both settings of the experiment however the increment in temperature at around time step 480 was better achieved by the model that was trained with some information from previous data of the temperature included. This was done because it was observed that temperature parameter is dependent on its own history in a way that could not easily be captured by working only from the pump voltage, pressure and RPM. The basic data for pump voltage and case pressure were somewhat noisy. This is of no consequence for the training process but the noise is inevitably transmitted

Table I
FIXED GP PARAMETERS USED FOR THE EXPERIMENTS

Runs	50
Population	1000
Generations	100
Crossover operator	Standard subtree crossover, probability 0.9
Mutation operator	Standard subtree mutation, probability 0.1,maximum depth of new tree 17
Tree initialisation	Ramped Half-and-Half, maximum depth 6
Function set	+, -, *, / log, exp
Terminal set	pump_v, press, rpm and random constants
Selection	Tournament of size 20
Elitism	Best individual always survives

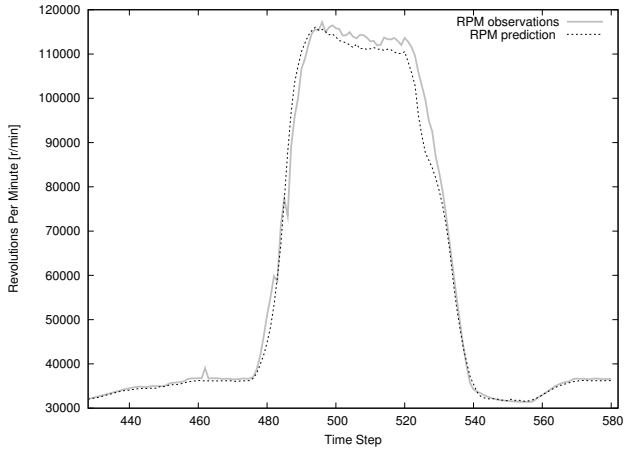

Figure 6. RPM prediction using pump voltage and pressure on test data

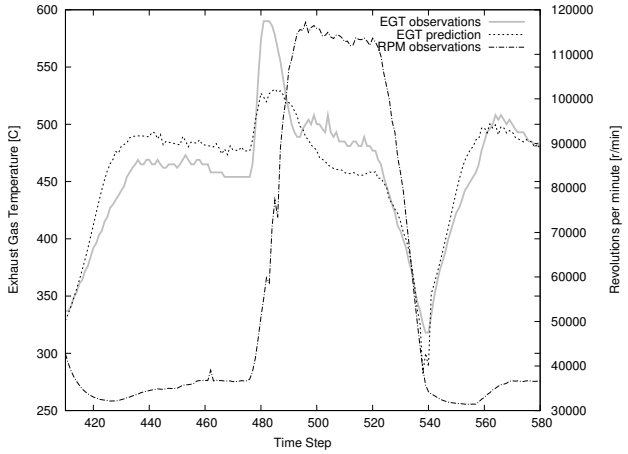

Figure 7. EGT prediction using pump voltage, pressure and rpm on test data. RPM observations in the same time period are also shown.

through the model. Therefore noise of the prediction has been reduced in both cases by using the pressure readings that had been pre-filtered during the data-logging process and by using the demanded voltage from the pump controller rather than the actual measured pump voltage.

Overall the results are good for predicted pressure and RPM. The predicted EGT is less accurate but still models the actual engine behaviour in a manner which is qualitatively correct. When the EGT history was not included as an input as in Figure 7 the transient peak in EGT during acceleration was not strong enough. The model that included EGT history, Figure 8 shows a much stronger peak but with some time lag. This suggests that further experimentation with the way in which EGT history is presented as an input may be required. Another approach would be to increase the prominence of the transient events within the training data so that these inaccuracies are penalised more strongly during the selection process.

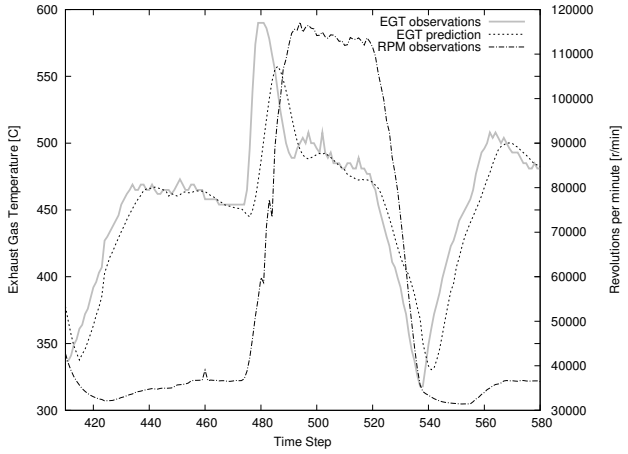

Figure 8. EGT prediction using pump voltage, pressure, rpm and temperature. RPM observations in the same time period are also shown.

V. CONCLUSIONS AND FUTURE WORK

In this paper an approach based on GP was presented to simulate different engine parameters for control design. A three stage simulation approach is proposed in order to predict the behavior of a jet engine. For this purpose, three models where created using GP. The first model

Table II
MODELS FOUND WITH THE BASIC GP IMPLEMENTATION

Predicted Variable	Mathematical Representation of the Model
Pressure	$pumpV_1/18.6264 + (pumpV_4 + ln(pumpV_2) + pumpV_2) * pumpV_1/18.6264$
RPM	$exp(2press_1 + 11.61046 + press_3/exp(press_3^2(press_3 * (press_1 + 5.80523) - press_3/$ $(press_3/(press_1 + 5.80523) + press_3/(press_3 * exp(pumpV_2) - press_1)))))$ $*92.3692 + press_3/exp(pumpV_2)) * 92.3692$
EGT	$6(pumpV_3/press_1) + 3(pumpV_3 * press_1) + temp_8 - pumpV_4 - ln(temp_{16})/pumpV_3 - 6ln(temp_{16})$ $+pumpV_3 + pumpV_3/press_1 + pumpV_3 * press_1$

uses pump voltage observations for pressure prediction. The second model uses pump voltage and pressure observations to predict the rotor speed. Finally the third model obtained uses the pump voltage, pressure and rotor speed to predict the exhaust gas temperature. Once the three models are obtained they can be used, one on top of the other, to have a complete simulation of all these variables through time, feeding the predicted values onto the models again to predict the parameters on the next time step. In general the three models were able to capture the relationship between variables. The exhaust gas temperature is less correlated than the rest of the parameters so the prediction was less accurate. To improve these results, further experimentation will be carried out increasing the number of historical values that are fed to the algorithm and by selection of the training data to provide a better balance between steady state and transient conditions.

The simulations thus derived can then be used to provide input to a further evolutionary process to produce an optimised controller for the engine. In the longer term it could be possible to integrate these algorithmic processes into the controller software itself to produce a controller that constantly optimises itself to account for changes in engine behaviour caused by climatic conditions and wear of components.

REFERENCES

[1] M. Bazazzadeh, H. Badihi, and A. Shahriari, *Gas Turbine Engine Control Design Using Fuzzy Logic and Neural Networks*, International Journal of Aerospace Engineering, Vol. 2011.

[2] M. Lichtsinder and Y. Levy, *Jet Engine Model for Control and Real-time Simulations*, Journal of Engineering for Gas Turbines and Power, Vol. 28, pp. 745-753, October, 2006.

[3] G. Koçer and O. Uzol, *Real-TIme Simulation of a Small Turbojet Engine*, 4. Ankara International Aerospace Conference, Ankara, 2007.

[4] R. Andoga, L. Madarasz and L. Fozo, *Digital Electronicl Control of a Small Turbojet Engine - MPM 20* 12. International Conference on Intelligent Engineering Systems, Miami, Florida, pp. 37-40, 2008.

[5] H. Gold and S. Rosenzweig, *A Method for Estimating Speed Response of Gas Turbine Engines*, NACA RM E51 K21, 1952.

[6] J.O.N. Lawrence and R.D. Powell, *The Application of Servo-Mechanism Analysis to Fuel Control Problems*, Proc. I. Mech. E., Vol. 172, pp. 439-469, 1958.

[7] V.L. Larrowe, M.M. Spencer and M. Tribus, *A Dynamic Performance Computer for Gas Turbine Engines*, Transactions of the ASME, Oct. 1957, pp. 1707-1714.

[8] S. M. Camporeale, B. Fortunato and M. Mastrovito, *A modular code for Real TIme Dynamic Simulation of Gas Turbines in Simulink*, Journal or Engineering for Gas Turbines and Power, Vol. 128, pp. 506-517, July 2006.

[9] Y. Yu, L. Chen, F. Sun and C. Wu, *Matlab/Simulink-Based Simulation for Digial Control System of Marien Three-Shaft Gas Turbine*, Applied Energy, Vol. 80, pp. 1-10. 2005.

[10] J. S. Litt, D. L. Simon, S. Carg, et. al. *A Survey on Intelligent Control and Health Management Technologies for Aircarft Propulsion Systems*, NASA, Glenn Research Center, Cleveland, Ohio, USA, 2005.

[11] F. Greitzer, et. al. *Gas Turbine Engine Health Monitoring and Prognostics*, SOLE '99 Symposium, Las Vegas, NV, September 1999.

[12] S. S. Tayarani-Bathaie, Z. N. Sadough Vanini and K. Khorasani, *Fault Detection of Gas Turbine Engines Using Dynamic Neural Networks*, 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Canada, 2012.

[13] H. Nayyeri and K. Khorasani, *Modeling Aircraft Jet Engine and System Identification by Using Genetic Programming*, 25th IEEE Canadian Conference on Electrical and Computer Engineering (CCECE), Canada, 2012.

[14] K. Schreckling, *Gas Turbine Engines for Model Aircraft*, Traplet Publications, ISBN 0951058916, 1994.

[15] Koza, J.R., *Genetic Programming: on the programming of computers by means of natural selection*, MIT Press, 1992.

[16] Poli, R. and Langdon, B. and McPhee, N. F., *A field guide to genetic programming with contributions by J. R. Koza*, Published via *http://lulu.com* and freely available at *http://www.gp-field-guide.org.uk*, 2008.

[17] Kotanchek, M. E. and Vladislavleva, E. Y. and Smits, G. F., *Genetic Programming Theory and Practice VII*, Springer US, 2010.

[18] Martinez-Arellano, Giovanna, Nolle, Lars and Bland, John. Improving WRF-ARW Wind Speed Predictions using Genetic Programming. SGAI Conf. 2012: 347-360.