ENGINE AIR PATH FAULT DIAGNOSIS USING ADAPTIVE NEURAL CLASSIFIER

M. S. Sangha, D. L. Yu, J. B. Gomm

Control Systems Research Group, School of Engineering,
Liverpool John Moores University, Byrom Street, Liverpool, L3 3AF, UK
Email: D.Yu@ljmu.ac.uk

Abstract: This paper presents a new method for on-board fault diagnosis for the air-path of spark ignition (SI) engines. The method uses an adaptive radial basis function (RBF) neural network to classify pre-defined possible faults from engine measurements to report the type and size of the fault. The RBF fault classifier adapts its widths and weights to model the time-varying dynamics of the engine and disturbance so that the false alarm rate is greatly reduced. The developed scheme is assessed with various faults simulated to a benchmark model and promising results are obtained. Copyright © 2006 USTARTH

Keywords: On-board fault diagnosis, automotive engines, adaptive neural networks, adaptive fault classification.

NOMENCLATURE

- t: time (sec)
- α: throttle plate angle (degrees)
- n: engine speed (rpm/1000)
- \( m_f \): engine port fuel mass flow (kg/sec)
- \( T_a \): ambient temperature (Kelvin)
- \( p_i \): absolute manifold pressure (bar)
- \( T_i \): intake manifold temperature (Kelvin)
- \( m_{at} \): air mass flow past throttle plate (kg/sec)
- \( T_{EGR} \): EGR temperature (Kelvin)
- \( m_{ap} \): air mass flow into intake port (kg/sec)
- \( m_{EGR} \): EGR mass flow (kg/sec)
- \( V_i \): manifold + port passage volume (m³)
- \( R \): gas constant (here 287 x 10⁻⁵)
- \( \kappa \): ratio of specific heats = 1.4 for air
- \( I \): crank shaft load inertia (kg m²)
- \( P_f \): friction power (kW)
- \( P_p \): load power (kW)
- \( P_p \): pumping power (kW)
- \( H_u \): fuel lower heating valve (kJ/kg)
- \( \Delta \tau_d \): injection torque delay time (sec)

1. INTRODUCTION

Fault detection, isolation and accommodation have become one of the most important aspects of automobile design. Continuous efforts are being made to improve the design of the Electronic Control Unit (ECU) in order to ensure early detection and isolation of engine faults, which can be hazardous for human life or lead to increased air pollution or adversely affect the fuel efficiency. OBD-II requires continuous monitoring and fault detection capability for all vehicle components whose failures can result in emission levels beyond 1.5 times of the Federal Test Procedure (FTP) standards. All vehicles sold in the UK after December 31st 2000 are required by legislation to allow the European On Board Diagnostic (EOBD) protocol.

A number of model-based fault detection and isolation (FDI) techniques (Frank et al. 2005) for automotive engines have been previously investigated. Artificial neural networks (ANN) are well known for good classification properties. Most ANN methods use a
fixed parameter network as the fault classifier. When the data is collected from the real engine to train the network, the classifier is able to cope with the model-plant mismatch caused by system complexity and non-linearity, compared with the model-based methods. However, after the classifier is put in real use for a period of time, engine dynamics change caused by mechanical ware of the parts, etc. and environment change will degrade the FDI performance of the classifier.

In this paper, a new on-line FDI scheme is proposed for engines using an adaptive neural network classifier. It has the following three features: (a) using the strong non-linear mapping (classifying) ability of the ANN to cope with the multivariable, severe non-linearity of engine dynamics; (b) the classifier is made adaptive to cope with the significant parameter uncertainty, disturbance and environment change; and (c) on-line fault diagnosis which can be directly implemented in an on-board diagnosis system (hardware). During operation, the network classifier learns parameter changes in the engine due to aging or environment change. It can also adapt to engine-to-engine differences within a batch of products. Gaussian radial basis function (RBF) neural nets are used for this purpose and both weights and widths are on-line adapted. Every sample of engine data is first tested for a fault and then used to update the neural network. The mean value engine model (MVEM) (Hendricks et al. 2004) and Capriglione, (2003). Details of the dynamics.

The manifold filling dynamics in reality is based on an adiabatic operation rather than isothermal. The manifold pressure can be represented by equation (1).

\[
\dot{p}_i = \frac{kR}{V_i} (-\dot{m}_\text{op} T_i + \dot{m}_\text{ad} T_a + \dot{m}_\text{EGR} T_{\text{EGR}})
\]

The positive terms within brackets show the in-flow of gas and the negative term shows the outflow of gas from the intake manifold (see Fig. 1). Using the law of energy conservation a state equation that describes the time development of the intake manifold temperature can be given as,

\[
\dot{T}_i = \frac{RT_i}{p_i V_i} \left[ -\dot{m}_\text{op} (k-1) T_i + \dot{m}_\text{ad} (k T_a - T_i) + \dot{m}_\text{EGR} (k T_{\text{EGR}} - T_i) \right]
\]

2.1.2 Crank Shaft Speed Dynamics

Applying the law of conservation of rotational energy, the crankshaft dynamics of an SI engine MVEM are described by equation (3).

\[
n = -\frac{1}{J_\text{m}^0} \left( P_f (n_i, n) + P_p (p_i, n) + P_b (n) \right) + \frac{1}{J_\text{m}^0} H_m \eta_\text{p}(p_i, n, \lambda) \dot{m}_f (t - \Delta \tau_d)
\]

where \( I \) is the scaled moment of inertia of the engine and its load and the mean injection/torque time delay has been taken into account with variable \( \Delta \tau_d \).

2.2 Fault Simulation

Four faults with four different fault intensities are considered for this research. Leakage in the intake manifold and EGR valve stuck up in different positions are simulated in the Simulink model as two component faults. The malfunction of manifold temperature and pressure sensors are simulated as two sensor faults. All the faults considered are realistic and have been considered by previous authors Nyberg and Stuttle, (2004) and Capriglione, et al. (2003). Details of the simulation of these faults are described as follows.

2.2.1 No Fault

For no fault situation, EGR is assumed to be 1/6 (16.67%) of the total air mass flow in the intake manifold. Practically EGR in a car can be as high as 20% of the total air mass flow. It is also assumed that all the sensors are working well and there is no leakage in the intake manifold. The no fault data is collected for different throttle angle inputs ranging from 20 to 40 degrees (the idling throttle angle for the engine is 15 degrees) for different operating points regarding load and speed.

2.2.2 Air Leakage Fault

To collect the engine data subjected to the air leakage fault, equation (1) is modified to
\[ \dot{p}_i = \frac{k_r}{V_i} \left( -m_{up}T_i + m_{at}T_a + \dot{m}_{EGR}T_{EGR} - \Delta I \right) \]  

(4)

where \( \Delta I \) is used to simulate the leakage from the air manifold, which is subtracted to increase the air outflow from the intake manifold. The air leakage levels are simulated as 5%, 10%, 15% and 20% of the total air intake in the intake manifold, respectively.

2.2.3 EGR Valve Faults

The normal value of EGR is about 16.67% of the total air mass flow and a realistic value of EGR feedback is chosen for the experiments. The value of \( \dot{m}_{EGR} \) for different fault intensities is regulated as 0%, 25%, 50%, 75% and 100% of the total EGR air mass flow. Where 0% EGR air mass flow corresponds to the EGR valve stuck up completely and 100% corresponds to full EGR air mass flow, i.e. no fault condition.

2.2.4 Temperature/Pressure Sensor Faults

Temperature and pressure sensor faults are considered in four different intensities: Sensors over-reading 20% or 10% and sensors under-reading 10% or 20% of the normal value. The faulty data for the sensors is generated using multiplying factors (MFs) of 1.2, 1.1, 0.9 and 0.8 for the above over- or under-reading respectively.

Faulty data are generated by the Modified MVEM with throttle angle at different values between 20° and 40° for all the fault conditions and no fault condition. The 17 states (4 faults * 4 intensities + 1 no-fault = 17) with their multiplying factors (MFs) are given in Table 1. The sample time is chosen as 0.1 sec. The engine data for the simulated faults and no fault condition covers almost all transient states of the engine dynamics.

<table>
<thead>
<tr>
<th>No</th>
<th>Fault Name</th>
<th>MF</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No Fault</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Leakage 5%</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Leakage 10%</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Leakage 15%</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Leakage 20%</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>EGR stuck 25% closed</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>EGR stuck 50% closed</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>EGR stuck 75% closed</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>EGR stuck 100% closed</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Temp. sensor 20% over reading</td>
<td>MF=1.2</td>
</tr>
<tr>
<td>11</td>
<td>Temp. sensor 10% over reading</td>
<td>MF=1.1</td>
</tr>
<tr>
<td>12</td>
<td>Temp. sensor 10% under reading</td>
<td>MF=0.9</td>
</tr>
<tr>
<td>13</td>
<td>Temp. sensor 20% under reading</td>
<td>MF=0.8</td>
</tr>
<tr>
<td>14</td>
<td>Pressure sensor 10% over read</td>
<td>MF=1.2</td>
</tr>
<tr>
<td>15</td>
<td>Pressure sensor 20% over read</td>
<td>MF=1.1</td>
</tr>
<tr>
<td>16</td>
<td>Pressure sensor 20% under read</td>
<td>MF=0.9</td>
</tr>
<tr>
<td>17</td>
<td>Pressure sensor 10% under read</td>
<td>MF=0.8</td>
</tr>
</tbody>
</table>

Table 1: The 17 fault and no fault states and multiplying factors

3. RBF NETWORK CLASSIFIER

A Gaussian RBF network is chosen in the research for the fault classification. There are several training algorithms. Those used for initial off-line training and on-line updating of the RBF classifier is presented.

3.1 The RBF Network Structure

The network consists of three layers; input, hidden and output layer. The input layer simply receives the network input vector \( x \in \mathbb{R}^d \), and passes the inputs to each node in the hidden layer. The hidden layer consists of \( n_h \) nodes that process the input vector. The \( i \)-th node in the hidden layer contains an individual centre vector \( c_i \) of the same dimension as \( x \) and a scalar width \( \rho_i \).

The Euclidean distance between the input and the centre vectors is calculated,

\[ z_i = \Vert x - c_i \Vert = \sqrt{(x_1 - c_{i1})^2 + ... + (x_n - c_{in})^2} \]

(5)

where \( i = 1, ..., n_h \), and passed through a non-linear basis function to produce the hidden node outputs \( \phi_i \). Several choices of basis function are available, e.g. thin plate spline, Gaussian function, etc. Gaussian basis functions provide a local excitation of the node with an output \( \phi_i \) near zero for inputs far from the centre and \( \phi_i \) near one for inputs close to the centre. This is especially suitable for classification applications and is therefore used in this work. The Gaussian basis function is defined as

\[ \phi_i = \exp\left[-\frac{1}{2}(z_i/\rho_i)^2\right], \quad \rho_i > 0 \]

(6)

Finally the network outputs are computed as a linear weighted sum of the hidden node outputs:

\[ y = W\phi \]

(7)

where \( y \in \mathbb{R}^q \) is the output vector, \( W \in \mathbb{R}^{q \times n_h} \) is the output layer weight matrix with element \( w_{ij} \) connecting the \( j \)-th hidden node to the \( i \)-th output, and \( \phi \) is a vector containing the hidden node outputs.

3.2 Training Algorithms

Training an RBF network means optimising the parameters of centres, widths and weights in the network. For off-line training, the K-means, the P-nearest neighbours and the batch least squares (BLS) algorithms are used. When the classifier is used on-line, the centres remain fixed, as they have been chosen distributed in the whole operating space, while the widths and weights are adapted to minimise the classification error caused by any time-varying dynamics and model uncertainty. The widths are adapted using a gradient descent algorithm and the weights are adapted using the recursive Least Squares (RLS) algorithm. These algorithms are reviewed or derived below.

(a) K-means Algorithm

The centres are set by the K-means clustering method whose objective is to minimise the sum squared distances from each input data to its closest centre so that the data is adequately covered by the activation functions \( \phi_i(t) \). The K-means clustering method is documented in Chen, et al. (1992).
(b) Gradient Descent method for the widths

When the RBF network is used to model a non-linear mapping or a dynamic system, the width in each hidden layer node is usually chosen as a constant using the P-nearest rule, or all widths are just chosen equal to the same value, as it is believed that the modelling error is not sensitive to the width. However, when the RBF network is used as a classifier, the classification is strongly sensitive to the Gaussian local function, which is mainly characterised by the width. Therefore, a gradient descent algorithm is derived to on-line adapt the widths to achieve a minimal objective function given as follows.

\[ J = \sum_{j=1}^{q} e_j^2 \quad (8) \]

where \( e_j = y_j - \hat{y}_j \) is the \( j \)th classifier output error and \( y_j \) is the \( j \)th training target. Then, according to the gradient method, the \( i \)th width can be adapted as

\[ \rho_i(k + 1) = \rho_i(k) - \alpha \frac{\partial J}{\partial \rho_i(k)}, \quad i = 1, \ldots, n_h \quad (9) \]

where \( \alpha \) is a learning factor and \( 0 < \alpha < 1 \). The gradient can be easily derived from equations (5)-(7) and (8),

\[ \frac{\partial J}{\partial \rho_i} = \sum_{j=1}^{p} \left[ \frac{\partial J}{\partial e_j} \cdot \frac{\partial e_j}{\partial \hat{y}_j} \cdot \frac{\partial \hat{y}_j}{\partial \phi_i} \cdot \frac{\partial \phi_i}{\partial \rho_i} \right] \quad (10) \]

and the four partial derivatives are derived and substituted into (10) giving,

\[ \frac{\partial J}{\partial \rho_i} = -4\phi_i \frac{||x - e||^2}{\rho_i^3} \sum_{j=1}^{p} e_j w_{ij} \]

Putting this back in equation (9), we have

\[ \rho_i(k + 1) = \rho_i(k) + 4\alpha \phi_i(k) \frac{|| \hat{X}_i - e ||^2}{\rho_i^3(k)} \sum_{j=1}^{p} e_j(k) w_{ij}(k) \quad (11) \]

This training algorithm can update the width parameter to minimise the sum of squared error defined in (8).

(c) Recursive Least Squares Algorithm

This algorithm is widely used for off-line training. If the RBF network has \( d \) inputs, \( q \) outputs and \( n_h \) hidden nodes, the output matrix with \( N \) samples (\( \hat{Y}^{N \times q} \)) can be written as

\[ \hat{Y} = \Phi(X) W \]

where \( X^{N \times d} \) is the input matrix, \( \Phi(X)^{N \times n_h} \) is the matrix of activation function outputs and \( W^{n_h \times q} \) is the matrix of weights. The RLS method is used for on-line training. It helps to construct on-line system models that should be more accurate and flexible than off-line fixed models. The RLS algorithm is given by Ljung (1999).

3.3 RBF Network Classifier

The RBF network, as the fault classifier will receive all possible and relevant signals containing fault information, and has 17 outputs with each indicating one of the investigated states in Table 1. The information flow for the fault diagnosis is illustrated in Fig.2.

Fig.2 Information flow of the fault diagnosis

According to the engine air path dynamics, four variables are chosen as the network inputs: the throttle angle, the manifold pressure, the manifold temperature and the crankshaft speed as shown in Fig.2. Two levels, 0 and 1, are used as the output targets of the classifier. Thus, the target matrix is a unity diagonal matrix of dimension 17 (when there is one training pattern for each fault) with each column being used as the classifier-training target vector. A successfully trained network will therefore diagnose the fault intensity as well as the fault type.

4. ON-LINE RBF CLASSIFIER ADAPTATION

The main contribution of this paper is that when the fault classifier diagnoses faults on-board, the classifier is adapted on-line so that the model-plant mismatch, parameter uncertainty and especially the time varying dynamics caused by mechanical wear of components and environment change can be modelled. In this way the classification error and consequently the false alarm rate can be greatly reduced. In fact, these effects are main drawbacks for the fixed parameter neural network to be used practically.

The fault classification and on-line adaptation are implemented as follows. Firstly, the measurements are read into the electronic control unit (ECU). Then, the data is fed into the classifier to diagnose faults. After this, the target will be modified according to whether a fault or several faults are detected. If a fault is detected the corresponding output of the target vector will be changed from “0” to “1”. Then the measurements and the modified target are used to update the classifier. In the adaptation, the width in each hidden node is adapted using the gradient descent algorithm in (11) and the centre location is remained fixed as previously described. This is followed by adaptation of the weights using the RLS algorithm.
5. SIMULATION & RESULTS

5.1 Data Collection & Normalisation

The simulation is run for different throttle angle inputs, 22°, 26°, 30°, 34° and 38° for no fault condition and data for manifold pressure, temperature and crankshaft speed is collected. In the same way the data for each fault condition listed in table 1 is collected for all the different throttle angle inputs. The simulation of different faults has already been explained in Section 2. A noise of normal distribution with zero mean and unity variance is then added to the collected data. The amplitude of the noise is set to about 2% of the average of the signal amplitude to simulate the measurement noise expected in a real engine system. The maximum noise amplitude considered is about 5%. To cover different engine working states, a number of data sets are collected for different initial and final values of the throttle angle as shown in Table 2.

Table 2: Details of data sets collected for training and testing of RBF networks

<table>
<thead>
<tr>
<th>Initial Throttle Angle</th>
<th>Final Throttle Angle</th>
</tr>
</thead>
<tbody>
<tr>
<td>Constant Speed</td>
<td>Acceleration</td>
</tr>
<tr>
<td>22</td>
<td>26, 30, 34, 38</td>
</tr>
<tr>
<td>26</td>
<td>30, 32, 34, 38</td>
</tr>
<tr>
<td>30</td>
<td>34, 38</td>
</tr>
<tr>
<td>34</td>
<td>38</td>
</tr>
<tr>
<td>38</td>
<td>38</td>
</tr>
</tbody>
</table>

Table 2 can be read in the way that for example, the first line has 5 data sets, one is with constant throttle angle and the other four with the throttle angle increased from 22 to 26, 30, 34 and 38 respectively. Every data set contained 340 samples, in each of which 20 samples were collected for no fault condition and for each fault condition. Three engine-operating modes when a fault occurs while driving a car on the road are considered:

(a) Car is running at a constant speed and a fault occurs
(b) Car is accelerating and a fault occurs
(c) Car is decelerating and a fault occurs

Total 27 different sets of such data are collected to ensure that all the possible combinations of engine operation, for different initial and final throttle angle values are taken into account, for the experiment within a particular range of engine operation. Initial training has been done before the on-line fault diagnosis with adaptive training.

The data is normalised by subtracting the steady state values and then scaled to the range of $[-1, 1]$, $x_{nor} = \frac{x - \bar{x}}{\sigma}$, where $x$, $\bar{x}$, $\sigma$ are measurements, their mean and standard deviation respectively. By normalisation there will not be any input data having a much greater numerical value than others and intending to dominate the training.

5.2 Network Training

The network input variables are chosen according to the experience in engine modelling as the four variables shown in Fig.2. The target matrix $X_o$ has ones in the first column up to the 20th row and all the other entries are zeros, the second column has ones from the 21st row to the 40th row, and so on. The last column has ones from the 321st row to the 340th row. This is shown as follows.

<table>
<thead>
<tr>
<th>Rows</th>
<th>X_o</th>
</tr>
</thead>
<tbody>
<tr>
<td>1~20</td>
<td>0</td>
</tr>
<tr>
<td>21~40</td>
<td>0</td>
</tr>
<tr>
<td>41~60</td>
<td>0</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>321~340</td>
<td>1</td>
</tr>
</tbody>
</table>

Thus, one of the 17 columns in the target matrix is associated to a fault condition. With the chosen input variables and the target, two RBF networks were trained with the training data set, where the same 20 centres chosen using the K-means clustering algorithm were used for the two networks. The widths were chosen using the p-nearest algorithm, and the weights were trained using the Batch Least Squares algorithm.

5.3 Fault Diagnosis

After training with the training data set, the two networks are used to diagnose faults with the test data set. The first network is used to classify the faults in the on-line mode with network adaptation. The forgetting factor for the RLS algorithm was chosen a constant value of $\lambda = 0.99$. The diagnosis result is shown in (f) – (h) of Fig.3. For comparison, the second network is trained off-line with the BLS algorithm and is not adapted on-line. This network is also used to classify the faults and the outputs are displayed in (a) – (e) in Fig.3. The diagnosis test result in Fig.3 (h) is also shown in Fig.4 with each output in a figure to display separately so that any misfire can be seen clearly. Fig.4 indicates that all the 17 outputs have correct fault detection results for the test data. It can be seen that the diagnosis by the adaptive network is much better than that by the fixed parameter network. The fault diagnosis result with the non-adaptive classifier in (a) – (e) have too much misclassification and is not successful except for (c) which is trained and tested for fixed throttle angle conditions (when the car is running on a constant speed and fault occurs). When the size of the non-adaptive network is increased to 200 centres chosen by the K-means method; the diagnosis result is shown in (b) of Fig.3. Though the performance is improved with an increase of centres, it still has many misclassifications and is still not as good as the adaptive method. In addition, 200 centres greatly increase the computing load and are more difficult to implement in practice. All the fault diagnosis is conducted with noisy data as explained in section 5.1.
It can be seen that these faults can still be classified when the data is contaminated by measurement noise. The classification with adaptive learning uses much smaller size of network and achieves a much higher successful rate. The on-line learning utilises the fault detection result so that faults are not learned as dynamics changes. Therefore, this scheme is still working after one fault occurs.

**Fig.3:** Fault classification test results with noisy data

**Fig.4:** All the 17 faults in Fig.3 (h) are shown separately for clarity.

### 6. CONCLUSIONS

A new adaptive RBF algorithm for on-board fault diagnosis of air path of an automotive engine is developed. The simulation results for a model with fixed parameter classifier and an adaptive parameter classifier are compared. The results of adaptive classifier are much better than non-adaptive classifier and are also better than classifier with much more hidden layer nodes. The adaptive algorithm is found to be tolerant to any parameter changes in the engine due to environment or aging. It is also independent of engine-to-engine changes caused by batch products. The developed method may not be sensitive to very slowly developed incipient faults, as these faults would be treated as system uncertainty and be learned by the network.

### 7. REFERENCES


