NEURAL COMPUTING
NCRG
RESEARCH GROUP

Neural Computing Research Group
Dept of Computer Science & Applied Mathematics
Aston University
Birmingham B4 7ET
United Kingdom
Tel: +44 (0)121 333 4631
Fax: +44 (0)121 333 4586
http://www.ncrg.aston.ac.uk/

Point-wise Confidence Interval Estimation by Neural Networks: A comparative study based on automotive engine calibration.

David Lowe
d.lowe@aston.ac.uk

Krzysztof Zapart

Technical Report NCRG/98/007 March 10, 1998

Submitted to Neural Computing and Applications

Abstract

In developing neural network techniques for real world applications it is still very rare to see estimates of confidence placed on the neural network predictions. This is a major deficiency, especially in safety-critical systems.
In this paper we explore three distinct methods of producing point-wise confidence intervals using neural networks. We compare and contrast Bayesian, Gaussian Process and Predictive error bars evaluated on real data.
The problem domain is concerned with the calibration of a real automotive engine management system for both air-fuel ratio determination and on-line ignition timing. This problem requires real-time control and is a good candidate for exploring the use of confidence predictions due to its safety-critical nature.

Keywords: neural networks, Gaussian Processes, confidence intervals, Bayesian error bars, safety-critical systems, engine management, air-fuel ratio, ignition timing

1 Introduction

The development of neural network approaches to real world problems has now reached a point where they are being seriously considered as candidates for providing the functionality for key embedded components in safety critical and mission critical systems. In these situations it is apparent that the ability to produce a forecast, regression or classification is not adequate without quantifying the *minimum* of additional knowledge that is required to determine the confidence intervals or error bars on the neural network outputs: it is necessary to quantify the *accuracy* or *reliability* of neural network performance: However there is no accepted 'standard' unique way to determine the neural network confidence intervals. The different approaches to evaluating error bars are based on different assumptions of model specification, data and noise distributions, and also have tradeoffs in terms of memory and speed of computation. In real world situations it is usually important to deliver information in a timely manner, even if that information is 'suboptimal'. Hence knowledge of 'the best' error bar may be useless if it takes too long to compute, or requires more input data than the overall system is capable of providing. In addition, in real systems where a neural network is but one small component, the overall system may demand only a reasonable level of tolerance of the function supplied by the neural network: a forecast which is only accurate to $\pm 10\%$ may be sufficient for some aspects of the overall system, but may be totally inadequate for a particularly key safety critical part of the system.

In this paper we investigate the error bar estimates produced by three different neural network motivated approaches. The problem domain considered is taken from a project concerned with providing automation and 'intelligent' control to internal combustion engine management systems. The specific example is a function mapping task where we wish to replace the standard look-up tables conventionally used to control the ignition timing and air-to-fuel ratios by a neural network which, in addition, provides a measure of confidence of its predictions. Performance is demonstrated both off-line (data collected from a test-rig engine under 'ideal' steady state conditions) and on-line (data collected directly from a vehicle driving around a typical urban environment and hence subject to transients). A neural network can produce much smoother control surfaces than linear look-up tables; moreover, a network can encode the optimum solution using a smaller amount of memory. In this problem domain the processing power in terms of CPU and memory is limited to the commercially available processors typically used in engine management systems. Such units usually contain only 256k of on-board memory. A neural network implementation, even with the additional functionality provided, would not be commercially viable if the solution required a modification to upgrade the processor, as this would also require a redesign of the complete engine management system. Hence this is a problem domain with commercial, engineering and technical constraints.

2 Confidence Intervals

In this paper we are concerned primarily with regression problems where the regression surfaces are multidimensional and nonlinear. Feed forward networks such as the multilayer perceptron and radial basis function network are ideal for this class of problem as they produce

a regression $y(\boldsymbol{x};\boldsymbol{\theta})$ conditioned on the input values $\boldsymbol{x}$ and the weights of the model $\boldsymbol{\theta}$. There has also been considerable recent work on attempting to produce self-consistent variance estimates around these predictions. For example, Tibshirani [12] considered the 'delta method' of error estimation based on the Hessian of the neural network, the 'sandwich estimator', and 'bootstrap' methods. The delta and sandwich methods are closely related and assume that the errors are i.i.d. random variables with a normal

distribution $N(0, \sigma^2)$, where σ^2 is estimated by maximum likelihood $\sigma^2 \propto \sum_{p=1}^{P} [t(\boldsymbol{x}_p) - y(\boldsymbol{x}_p; \boldsymbol{\theta})]^2$. From a Taylor expansion of the log likelihood function, an estimate of the standard error about the prediction is determined by the inverse of the 'observed information matrix', I - the negative of the Hessian of the neural network. This is related to 'Bayesian error bars' discussed further below. Tibshirani also considered a bootstrap approach to error estimation in which many pseudo replicates of the training set are produced. Weight parameters are determined for each of these bootstrap samples. These represent samples from possible model space, each of which gives a slightly different prediction around the training samples and hence error estimates can be obtained. However this is an unrealistic approach in a practical situation as it requires many replicates of the data, each of which requires an optimisation of the model complexity and weights. It is infeasible as a model for novel data or on-line confidence interval estimation in our problem domain, which is unfortunate as Tibshirani concluded that the bootstrap provided the best estimates of those models considered.

Kim and Bartlett [3] developed an alternative and more efficient approach to confidence interval estimation, by introducing a supplementary neural network trained to predict the error of the primary network by taking as inputs both the inputs and outputs of the primary network. In their approach the first primary network is trained on the task, producing an output, $\boldsymbol{y}(\boldsymbol{x}; \boldsymbol{\theta})$ and a corresponding error signal $\boldsymbol{\epsilon}(\boldsymbol{x}; \boldsymbol{\theta}) = |t(\boldsymbol{x}_p) - y(\boldsymbol{x}_p; \boldsymbol{\theta})|$. The supplementary network takes as input, the input vector $\boldsymbol{x}$, the output vector of the primary network $\boldsymbol{y}(\boldsymbol{x}; \boldsymbol{\theta})$ and tries to predict the corresponding error term $\boldsymbol{\epsilon}$. Since the output of the main network is fed directly into the supplementary network, Kim and Bartlett termed this approach 'error estimation by series association'. The predictive error bars advocated later in this paper is similar in spirit though distinct in architecture to this 'series association' method.

Although there are several other approaches to estimating pointwise confidence intervals, we now describe in detail the three basic models that are likely to be most suitable for providing confidence interval estimation on the specific problem of engine calibration.

2.1 Bayesian error bars

In the Bayesian approach to error modelling discussed in [13] there are basically two sources of error: the first is concerned with the intrinsic noise on the *target* data, which is constant, and the second is a consequence of the error on the weights themselves. These two terms are independent and can be combined to produce the total output error:

$$\sigma_y^2(\boldsymbol{x}) = \sigma_w^2(\boldsymbol{x}) + \sigma_t^2(\boldsymbol{x}), \tag{1}$$

where $\sigma_w^2(\boldsymbol{x})$ is the variance of the output due to weights uncertainty, $\sigma_t^2(\boldsymbol{x})$ is the variance of the data noise and $\sigma_y^2(\boldsymbol{x})$ is the overall output variance.

Under a Gaussian approximation to the posterior weights distribution the first term may be approximated by

$$\sigma_w^2(\boldsymbol{x}) = \boldsymbol{g}^T(\boldsymbol{x}) \boldsymbol{H}^{-1} \boldsymbol{g}(\boldsymbol{x}), \tag{2}$$

where $\boldsymbol{g}(\boldsymbol{x})$ is the weight gradient of the neural network output and $\boldsymbol{H}$ is the Hessian matrix of the model.

The noise variance $\sigma_t^2(\boldsymbol{x})$ is usually assumed to be constant and approximated by

$$\sigma_t^2(\boldsymbol{x}) = \frac{1}{\beta} = \frac{2E_D}{(N - \gamma)}, \tag{3}$$

where N is the number of training examples, γ is the number of well-determined parameters in the model (e.g. weights in a neural network) and E_D is the error measured on the training set. The

parameter γ can either be approximated by the number of weights k in the models or, according to the full Bayesian treatment of the error bars, it could be set to

$$\gamma = k - \alpha \mathrm{Trace}(\boldsymbol{H}^{-1}), \tag{4}$$

where α is a regularisation parameter.

Also, when implementing the full Bayesian approach, the training should be iterated until the values of hyperparameters α and β converge. This, however, significantly slows down training of the networks since, at each iteration, the Hessian matrix needs to be evaluated and inverted, and the training is repeated with pseudo-inverting the design matrix (in the case of RBF networks) each time.

On the other hand, the average number of iterations needed for the

hyperparameters to converge is not very large, typically being only

four or five.

In this paper we concentrate upon Radial Basis Function networks. For this case the calculation of the approximate Bayesian error bars is simplified since the Hessian matrix is given by an exact formula [13]

$$\boldsymbol{H} = \beta \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \alpha \boldsymbol{I}, \tag{5}$$

where

$$\alpha = \frac{\gamma}{2E_W}, \tag{6}$$

$$E_W = \frac{1}{2}\boldsymbol{w}^T \boldsymbol{w}, \tag{7}$$

$\boldsymbol{w}$ is the weight vector and $\boldsymbol{\Phi}$ is the design matrix for a given Radial Basis Function network.

Note that a major assumption in the Bayesian error bars approach (apart from assumptions of peaked Gaussian posterior parameter distributions allowing linearisation) is that it is predominantly driven by the density of input data. Although it also depends on the global training error, this relation is not localised. In some regions both the training error and generalisation error may be low (hence the network should be very confident), but still the error bars may be unreasonably high as a consequence of large noise processes elsewhere in data space.

Nevertheless Bayesian error bars can indicate that in the regions where the input data is scarce the network predictions may have a larger error. This is qualitative information which can be used for gathering further data points in these high-error/low data density regions.

Another major restriction is the assumption of constant noise variance on the targets. The assumption is useful since it simplifies the analysis and reduces the computation required. However there may be examples when the assumption introduces discrepancies when compared to the predictions of a more complete model. The consequences of this assumption in practice will be exhibited later.

Recent work has attempted to extend the model by allowing for noise dependent inputs [9].

2.2 Gaussian Processes

The second approach to error bar estimation that we will investigate in this paper is the Gaussian Process model, which is disscussed in [14]. This approach can be considered to be equivalent to a

neural network in the limit of an infinite number of hidden units assuming Gaussian weight priors. The Gaussian Process model assumes a stochastic source characterised by a mean and covariance only and no higher moments are necessary. In general the predictive distribution is Gaussian with mean and variance given by

$$\begin{aligned} y_G(\boldsymbol{x}) &= \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{t} \\ \sigma_y^2(\boldsymbol{x}) &= C(\boldsymbol{x}, \boldsymbol{x}) - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k} \end{aligned} \tag{8}$$

where $\{\boldsymbol{x}_1, \ldots \boldsymbol{x}_P\}$ is the set of training examples,

$\boldsymbol{t}$ is the set of training targets,

$\boldsymbol{k} = (C(\boldsymbol{x}, \boldsymbol{x}_1), \ldots C(\boldsymbol{x}, \boldsymbol{x}_P)^T$ and $\boldsymbol{K}$ is the covariance matrix for training cases. Hence the error bars are explicit by the predictive distribution being Gaussian.

One major flaw with this approach is that it does not scale well, due to the requirement of inverting a $P \times P$ covariance matrix. This probably excludes this model from being a practical contender for implementation in the current problem. However its explicit construction of the variance around the prediction makes it an ideal model for benchmarking.

2.3 Predictive Error bars

The third approach to error bar estimation by neural networks is what we shall term *predictive error bar estimation*. In this approach we construct a neural network model of the regression on the conditional target variance to model the noise process. It is an approach suggested by [10] in a neural network context. In a non neural network context in econometric time series modelling, a similar method but employing a Kalman Filter as the noise model is known as *prediction error variance* [2].

This approach is based on the fact that for a network trained on minimum square error, the optimum network output approximates the conditional mean of the target data, $y_{opt}(\boldsymbol{x}) \approx \langle \boldsymbol{t}(\boldsymbol{x}) | \boldsymbol{x} \rangle$. Hence for each input pattern $\boldsymbol{x}$ we may estimate the local variance as $||\boldsymbol{t}(\boldsymbol{x}) - y_{opt}(\boldsymbol{x})||^2$. If this local variance is used as the training target for another neural network, then the optimum output of this second network is just the conditional target value again. However for these modified targets, this optimum output is therefore an approximation to the local expected variance which we interpret as a confidence interval

$$\boldsymbol{\sigma}^2(\boldsymbol{x}) = \langle ||\boldsymbol{t}(\boldsymbol{x}) - \langle \boldsymbol{t}(\boldsymbol{x}) | \boldsymbol{x} \rangle||^2 | \boldsymbol{x} \rangle \tag{9}$$

In the implementation of predictive error bars there are two interlocked neural networks. Each network shares the same input and hidden nodes, but has different final layer links to separate output nodes. One set produces the required regression to the desired target values whereas the second one (the error network) approximates the residual error surface of the first model. This surface is extracted from the first network by measuring the residual error on the training set (target values are known). Thus the second network predicts the noise variance of the main neural model. The approach readily extends to committees of networks. Figure 1 depicts the interlocking network architecture.

Optimisation of the weights is a two-stage process: the first stage determines the weights $\boldsymbol{w}_1$ conditioning the regression on the mapping surface. Once these weights have been determined, the network approximations to the target values are known, and hence so are the conditional error values on the training samples. In the second stage, the inputs to the network remain exactly as before, but now the target outputs of the network are the conditional error values. This second pass determines the weights $\boldsymbol{w}_2$ which condition the second set of output nodes to the (squared) error

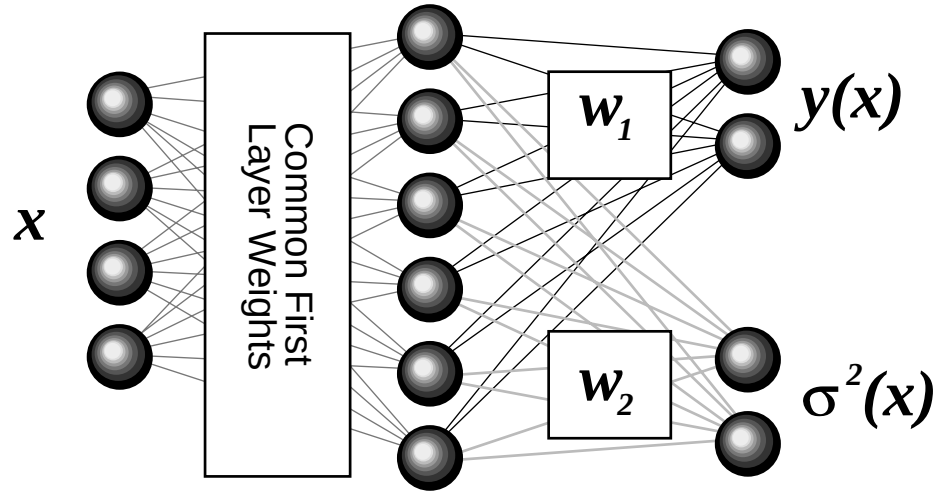

Figure 1: The architecture of the predictive error network. The second interlocking network which produces the confidence prediction is displayed in a lighter shade of grey.

values $\boldsymbol{\sigma}^2(\boldsymbol{x})$. Note that it is reasonable to employ the same hidden units for both the regression and the error networks, since the first layer determines the unconditional distribution of the data - or extracts the relevant feature space of the problem - if the first layer weights are optimised. Hence the error network is modelling the same data distribution as the regression network. It also reduces training time if the first layer weights are optimised as the optimisation of the first layer weights is performed only in the first pass phase.

This approach is computationally fast, does not increase the input dimensionality of the network as in some other approaches to confidence interval estimation, and does not rely on the smoothness and peaked Gaussian assumptions of the Bayesian approach. However it does rely on the ability of the neural network to approximate the conditional regression correctly. Hence model complexity and the bias-variance dilemma are issues which need to be addressed. Also note that if we make the additional assumptions of a Gaussian noise distribution, then we can write down a likelihood model, the maximum of which corresponds to the predictive error estimate. For this reason, predictive error bars are also sometimes known as maximum likelihood error bars. However the assumption of a Gaussian model is not essential and hence predictive error bars have a wider generality. In particular note that predictive error bars are unconstrained estimates. Since they are not based on a tightly constraining Gaussian model they are not parameterised to automatically have the correct properties of a variance estimate. In particular, in situations where a network is overtrained and then is used to extrapolate, then there is no guarantee that the confidence interval will remain positive. In such situations, higher order uncertainty is required (confidence in the confidence estimates). Another way of

circumventing this problem is to build in constraints such as a possibility of an exponential function

on the output.

In terms of computational complexity the predictive error bars would be most efficient to implement, followed by Bayesian error bars and Gaussian Process error bars. We now demonstrate the behaviour of these different estimates of confidence, beginning with a simple illustrative example.

3 Synthetic Example

As a simple example of the behaviour of the three different approaches to estimating pointwise confidence, consider the example of data generated according to a simple sine wave. The problem is slightly complicated by having a higher data density near to the origin. In addition the raw samples have a space-dependent additive noise process applied. The noise is randomly uniform with a maximum absolute spread of 0.7 and is only effective in the region $[\pi/2, 3\pi/2]$ and is zero elsewhere.

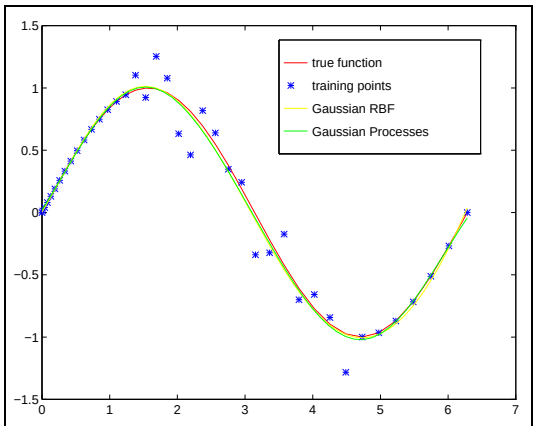

Figure 2: Synthetic signal with nonuniform distribution of samples.

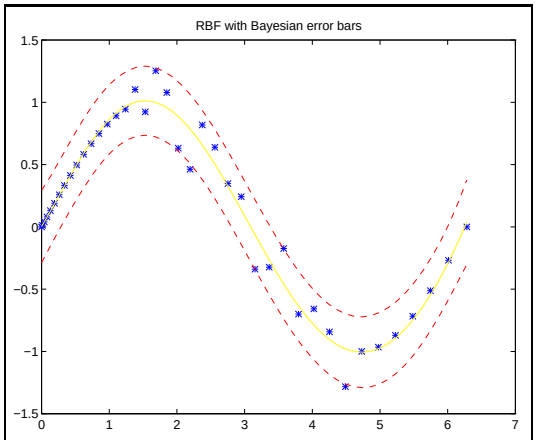

Figure 3: Radial Basis Function network with Bayesian error bars

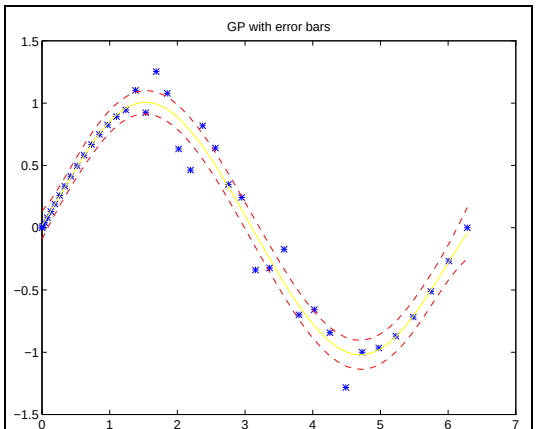

Figure 4: Gaussian Process model predictions with error bars

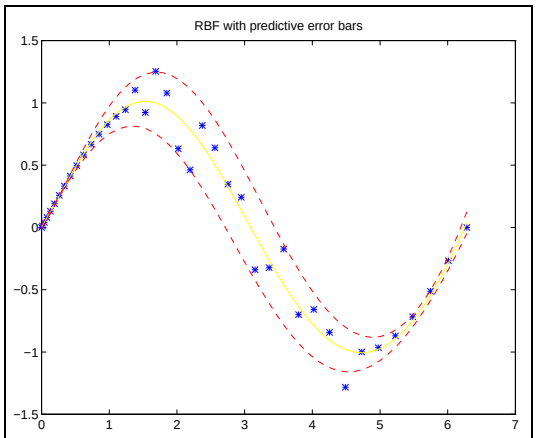

Figure 5: Radial Basis Function model with predictive error bars

Figure 2 depicts the true generator, and the training data samples used. The additional figures (Figures 3, 4, 5) show the results of the regression curves plus the calculated confidence intervals

displayed as 2σ curves around the local predicted means. Note that all models produce good regression curves, but yield different behaviour of the confidence intervals.

The Gaussian process error bars are unrealistically small in regions of high noise and in addition require significant computation time. The Bayesian error bars reflect a sensible dispersion in the region of uniform noise, and also indicate broad error bars in the region of low data density. However, it also displays large errors in regions of high data density. This is a consequence of the limitation of assuming a constant noise variance on the targets. Although work is in progress to relax this assumption [9] to allow position dependent noise variance, this extended approach requires more data and significantly more computational time to produce the error bars. In contrast, the predictive error bars require least processing time, exhibit large and approximately uniform error bars in the region of uniform noise, and reduced error bars in the region of zero noise and low data density.

The inability of the Bayesian approach to deal with position dependent noise variance is a significant disadvantage in real world situations in which we are more interested.

4 Neural Networks Applied to Automotive Engine Calibration

Applying neural network methods to vehicle engine management systems is not novel [4; 5; 6; 7; 8; 11]. However none of these approaches have considered the consequence of less than perfect neural network predictions. We now demonstrate the behaviour of the selected error bar models on one small aspect of automatic engine control.

Currently, modern engine management systems regulate functions such as the ignition timing and the air-to-fuel ratio of the combustion mixture with piecewise linear interpolation look-up tables based on calibration points. We are interested in replacing these by full regression surfaces for the ignition timing and the air to fuel ratio (AFR) maps. In normal engine operation (away from idle speed) the ignition timing and fuel injection volume is determined by recourse to a set of human-derived look up tables as a function of several variables such as load, speed, engine temperature etc. These look up tables are obtained from laborious and human-intensive experiments and are typically produced as selected values in a sparse matrix, typically of size 16×16 cells or less. There are many such look up tables in modern engine management systems governing all aspects of an engine operating envelope. The criteria for 'correct' values are complex, involving tradeoffs between performance, exhaust emissions, economy and driveability.

As part of a process of developing neural network methods to engine management systems we have constructed replacements for some of these look up tables using a neural network. This has the advantage of being efficient in processing power, capable of on-line adjustment due to engine aging, but mainly it has the ability to produce continuously analog output values, rather than being restricted to the values of the look up table cells. In addition the approach allows us to produce estimates of confidences on the engine maps which could be used as part of an overall safety critical assessment system. This real world problem is characterised by a sparsity of data and a significant expense of collecting each extra data point. Hence it is usual for only part of the look up table to be determined by experiment and the remaining cells specified by 'expert knowledge' of the engineer – especially on the boundaries of the map.

There are two different sources of errors in making predictions. Firstly, because these maps have been produced by engineers, it is considered that the values in the map are not very accurate and hence the maps contain some errors due to a human factor. The second source of the errors in

predictions should be due to sparsity of data and inaccuracies in the neural network models. As the human errors cannot be accurately captured (or corrected) by a neural network it is considered that the values in the map are reasonably accurate and hence the target signal to noise ratio is high. So the dominant sources of the error in predictions should be due to problems with sparsity of data and neural network representaions of the data. We will see the effect of this in the error bar plots next.

We begin by concentrating only upon the AFR maps. The ignition timing maps are similar, but because the AFR maps tend to be more variable the forecasts on the ignition timing maps tend to be more accurate with smaller error bars. Hence modelling the AFR maps tends to provide a better comparison of the approaches.

4.1 AFR mapping

Figure 6 shows the actual data in a real AFR map for an engine as a function of just the load and speed of a vehicle. For this experiment the original map consists of 256 samples in the 16×16 map. Some of these values were determined by experiment, and the others were specified by prior expert domain knowledge. As a consequence of having mixed sources of data, we would expect there to be natural variations in reliability in the actual data itself, particularly at boundaries and in regions of rapid variations of the maps where it would be difficult to maintain precise load, speed and AFR values.

The error surfaces obtained by optimising a Radial Basis Function network and Gaussian process models to produce a regression surface to Figure 6 are displayed in figures 7–9. We note that the Bayesian and Gaussian error surfaces are more uniform than the predictive error surface, despite the fact that there is a nonuniform distribution of error in the data. The predictive error bars are more sensitive to regions in the map which are subject to significant changes. This reflects the reality in that it is more difficult to determine the correct AFR values in operating envelope regions which are changing rapidly.

On the basis of these results we conclude that the predictive error bar model is both efficient and effective. In a real implementation it is likely that on criteria of low computational requirement combined with the good overall performance, the predictive error bar would be the chosen implementation.

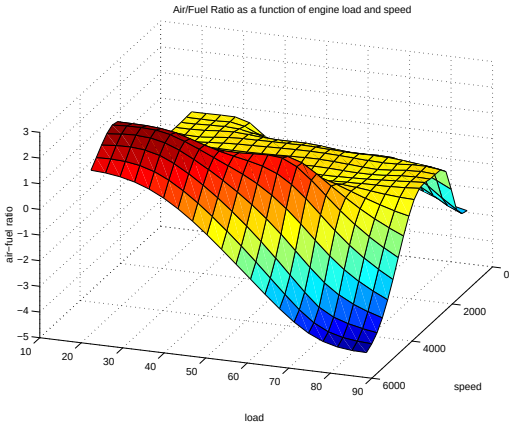

Figure 6: AFR raw data: 256 uniformly distributed samples an a 16×16 grid

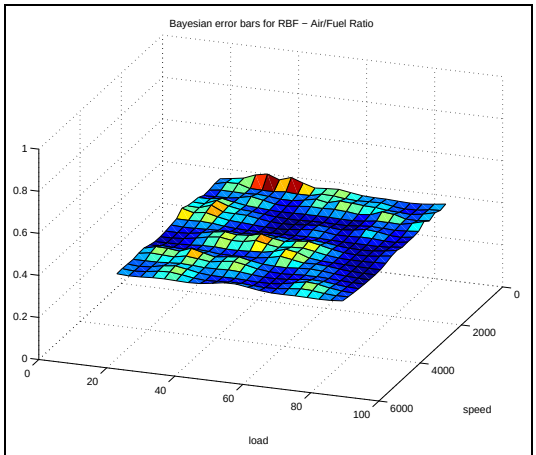

Figure 7: Bayesian error bars produced by a Radial Basis Function regression

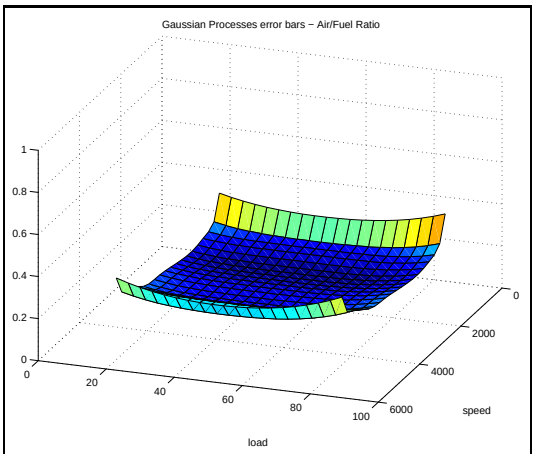

Figure 8: Error surface produced by the Gaussian Process regression model.

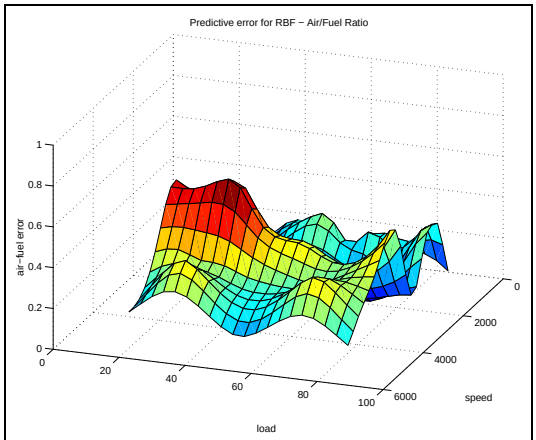

Figure 9: Predictive error bars produced by a Radial Basis Function regression

4.2 On–line behaviour

The previous experiments were off-line regression problems where the neural network was used to mimic the behaviour of the expert engineer in producing AFR maps. In this off-line experiment we do not have access to intermediate values employed by the real engine. Since the neural network gave predictions at all intermediate values of load and speed sites it is interesting to speculate whether the neural network generalisation performance extends to on-line performance as well. In this final experiment we examine on-line ignition timing performance using just the radial basis function network employing predictive error bar estimation.

An instrumented vehicle was driven around an urban route under normal conditions, including steady state and transient, under a variety of load/speed conditions. The actual ignition timing employed by the engine was logged, along with other details such as the estimated load and measured speed. This actual performance was then compared to the predicted ignition timing values as produced by an appropriate radial basis function network. The results of this on–road test comparing the actual engine performance and the model are depicted in Figure 10.

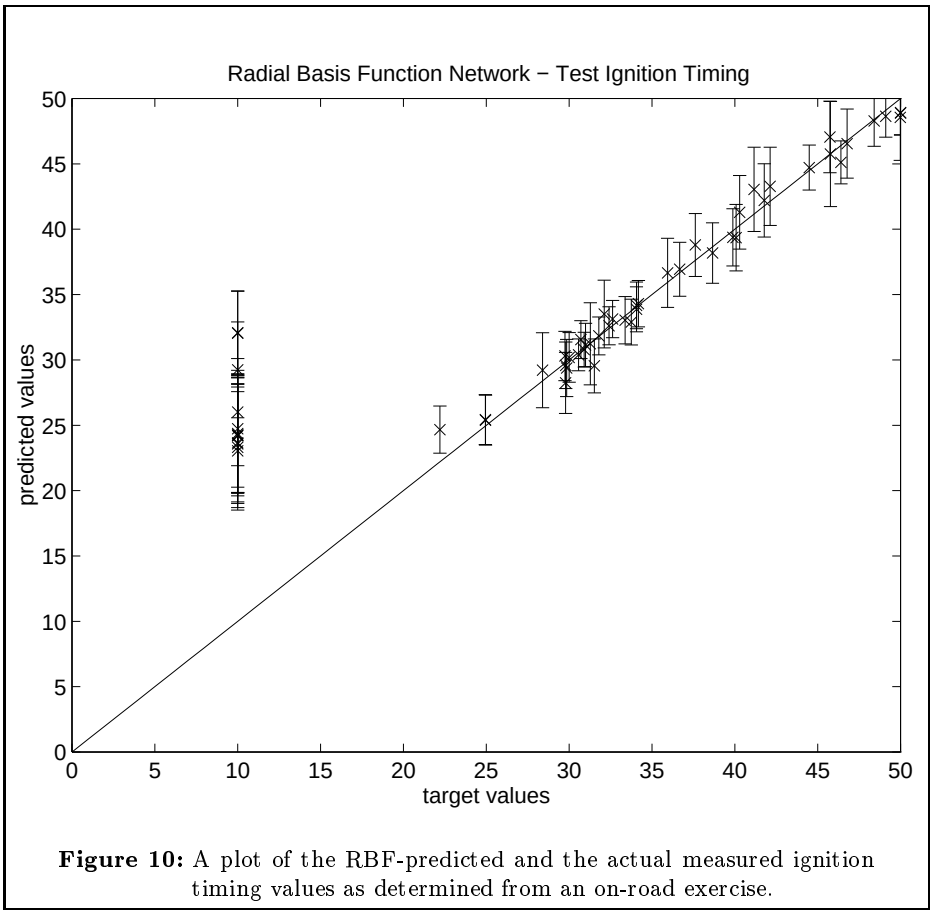

Figure 10: A plot of the RBF-predicted and the actual measured ignition timing values as determined from an on-road exercise.

This figure also displays the calculated *predictive error bars* of the Radial Basis Function model which are superimposed as 2σ bounds around the predicted mean. It is noted that there is exceptionally good correlation between the predictions and the actual values, especially when the confidence intervals are taken into account. The one exception is a region on the graph which corresponds to the idle speed conditions of the vehicle. In the actual engine, idle speed conditions are accounted for by an entirely separate map for ignition timing which the controller switches over to when load-speed conditions drop below a threshold. No allowance was made for the idle speed map by the neural network regression in that no training data points were employed corresponding to

these extreme conditions. In this sense, idle speed conditions are not representative of the training data available to the original network model.

5 Conclusion

We have considered the implication of employing neural networks in safety critical real world situations from the perspective of developing confidence intervals on neural network predictions. This is not trivial as there are several approaches one may adopt in constructing confidence intervals, as discussed. In the problem considered in this paper it is important to obtain an estimate of the mapping surface and a level of confidence in real time (determined by the cylinder firing rate and hence is down at the tens of millisecond time frame). We have found that an approach which we refer to as 'predictive error bars' are more suited to data noise and also are more efficient in terms of computational power required, which is a constraint in current engine management systems based on conventional EEPROM controllers. Although this paper has only demonstrated results on individual neural network models, the approaches outlined in the paper have also been extended to *committees of networks* involving combinations of radial basis function networks, multilayer perceptron networks and linear models. This has the advantage of improving regression performance by combining different failure modes to reduce the error bars. In principle this should be a more robust approach for safety critical systems.

Finally, one remaining issue not discussed in this paper but left as an open question is to pose the challenge of how this extra 'confidence' information provided by a neural network can be exploited in an embedded system to determine system–level validation and verification.

Acknowledgements:

This work was supported by SAGEM and under EPSRC contract K51792. The authors would like to thank Hector Sindano and the staff of SAGEM Ltd for the provision of data and background information on the problem domains.

References

[1] L A Feldkamp, G V Puskorius, L I Davis and F Yuan, "Neural control systems trained by dynamic gradient methods for automotive applications", *Int. Joint Conference on Neural Networks*, **2**, 798–804, (1992).

[2] A C Harvey, "Time Series Models", Philip Allan Publishers, Oxford, (1981).

[3] K. Kim and E. B. Bartlett, "Error Estimation by Series Association for Neural Networks", *Neural Computation*,**7**(1), 799–808, (1995).

[4] S. Leonhardt, H. Gao, V. Kecman, "Real Time Supervision of Diesel Engine Injection with RBF-based Neural Networks", *Proceedings of the American Control Conference, Seattle, Washington*, 2128–2132, (1995).

[5] S Morita, "Optimization control for combustion parameters of petrol engines using neural networks — in the case of on–line control", *Int J of Vehicle Design*, **14**(5/6), 552–562, (1993).

[6] P. O'Reilly, S. Thompson, "A neural network air-fuel estimator", Proceedings of the IEE Control Conference, 165–172, (1994).

[7] G. V. Puskorius, L. A. Feldkamp, "Automotive Engine Idle Speed Control with Recurrent Neural Networks", *Proceedings of the American Control Conference, San Francisco, California*, 311–316, (1993).

[8] G V Puskorius, L A Feldkamp and L I Davis, "Dynamic Neural Network Methods Applied to On-Vehicle Idle Speed Control" *Proc IEEE*, **84**,(10), 1407–1420, (1996).

[9] C S Qazaz, "Bayesian Error Bars for Regression", PhD Thesis, Aston University, 1996.

[10] C Satchwell, "Finding Error Bars (the easy way)", *Neural Computing Applications Forum*, Edition 5, (1994).

[11] S P Stevens and T H Ma, "Experimental data processing techniques to map the performance of a spark ignition engine", *Proc. Inst. Mech. Engrs*,**209**,297–306, (1995).

[12] R. Tibshirani, "A comparison of some Error Estimates for Neural Network Models", *Neural Computation*, **8**(1),pp 152–163, (1996).

[13] C. K. I. Williams, C. Qazaz, C. M. Bishop and H. Zhu, "On the relationship between Bayesian error bars and the input data density", *Proceedings of the Fourth International Conference on Artificial Neural Networks*, Cambridge, U.K., 160–165, (1995).

[14] C. K. I. Williams, C. E. Ramussen, "Gaussian Processes for Regression", *Advances in Neural Information Processing Systems*, **8**, 514–520, (1996).