Scientific Research Publishing

Journal of Software Engineering and Applications, 2020, 13, 327-347
https://www.scirp.org/journal/jsea
ISSN Online: 1945-3124
ISSN Print: 1945-3116

Sensitivity Analysis of Radial Basis Function Networks for River Stage Forecasting

Christian Walker Dawson

Department of Computer Science, Loughborough University, Leicestershire, UK
Email: c.w.dawson1@lboro.ac.uk

How to cite this paper: Dawson, C.W. (2020) Sensitivity Analysis of Radial Basis Function Networks for River Stage Forecasting. *Journal of Software Engineering and Applications*, **13**, 327-347.
https://doi.org/10.4236/jsea.2020.1312022

Received: October 21, 2020
Accepted: December 20, 2020
Published: December 23, 2020

Abstract

Sensitivity analysis of neural networks to input variation is an important research area as it goes some way to addressing the criticisms of their black-box behaviour. Such analysis of RBFNs for hydrological modelling has previously been limited to exploring perturbations to both inputs and connecting weights. In this paper, the backward chaining rule that has been used for sensitivity analysis of MLPs, is applied to RBFNs and it is shown how such analysis can provide insight into physical relationships. A trigonometric example is first presented to show the effectiveness and accuracy of this approach for first order derivatives alongside a comparison of the results with an equivalent MLP. The paper presents a real-world application in the modelling of river stage shows the importance of such approaches helping to justify and select such models.

Keywords

Artificial Neural Networks, Backward Chaining, Multi-Layer Perceptron, Partial Derivative, Radial Basis Function, Sensitivity Analysis, River Stage Forecasting

1. Introduction

Rapid developments in artificial intelligence in recent years, and ever more powerful computing resources, have led to a plethora of machine learning techniques being used to solve all kinds of hydrological problems—for example, multi-layer perceptrons, radial basis function networks, support vector machines, deep learning, etc. Numerous studies have compared different techniques in different regions for different hydrological problems (for example, [1] [2] [3] [4]). The bulk of the research has involved the use of artificial neural networks (ANNs) including the popular multi-layer perceptron (MLP).

Although MLPs are arguably the most recognised feed forward neural net-

work models, Radial Basis Function networks (RBFNs) have proved to be a popular alternative as they can be trained relatively quickly and perform comparably well against equivalent MLPs. RBFNs [5] [6] [7] were developed from an exact multivariate function interpolation [8] and have attracted a lot of interest since their conception in the late 1980s. Although there are differences in the transfer functions, how these functions combine and how they are trained, the conceptual structure of MLPs and RBFNs is equivalent—consisting of several interconnected, layered nodes in a feed forward structure. Thus, RBFNs and MLPs are both classed as feed forward neural networks (FFNNs).

However, with such a diverse choice of tools open to the hydrologist, the challenge of deciding which technique to use can be daunting. The choice of which machine learning technique to use can be broken down into three fundamental issues:

1) Difficulty—how difficult is it to calibrate the chosen model—how long will it take; and how complex is the solution?

2) Accuracy—how accurate is the model we have produced?

3) Confidence—how confident are we that the model has some physical interpretability of its behaviour so we believe in the predictions it is making and can justify these predictions in a rational sense?

While studies have shown the MLP and RBFN can address the difficulty and accuracy issues defined above (they are relatively easy to implement and understand; and studies have shown them to be accurate), there has been little work addressing the third concern—that of physical interpretability. The objective of this paper is thus to contrast these two common machine learning variants within this context by exploring an approach to sensitivity analysis which provides some physical interpretation.

Sensitivity analysis of FFNNs is an important research issue as it allows us to explore relationships, provide physical and/or mechanical rationality in certain practical applications and justify model behaviour. Chen [9] identified several approaches in the literature to sensitivity analysis of FFNNs. For example, one such FFNN is Madaline—developed in 1988 by Winter and Widrow. A number of approaches have been used to explore the sensitivity of Madaline Neural Networks including Stevenson *et al.* [10] who derived sensitivity from a percentage error of the network's weights; Piche [11], based on a statistical model; and Zeng *et al.* [12] who applied a hypercube probability model. In terms of MLPs, a number of approaches have been explored including Hashem [13] who applied backward chaining to derive the partial derivatives and hence sensitivity of the model to each input; Yang *et al.* [14] who considered sensitivity of MLP outputs with respect to input perturbations; and Zeng *et al.* [15] who employed a similar technique that also perturbed connection weights. Binary neural networks have also attracted some attention in terms of sensitivity analysis. For example, Huang *et al.* [16] used matrix and probability theory to determine a network's sensitivity which was less computationally demanding than previous approaches.

The RBFN has also attracted interest in terms of sensitivity analysis—most notably by Ng *et al.* [17] who looked at output changes based on input perturbations; Shi *et al.* [18] who examined squared output deviations; and Chen *et al.* [9] who defined a quantified measure of sensitivity based on input perturbations.

This paper applies the backward chaining rule of Hashem [13] to derive partial derivatives of the network output with respect to each input for RBFNs consisting of a linear output unit and Gaussian basis functions. Although the partial derivative of RBFNs has been defined and used before (for example, [19] [20]), no study has been made into such an application in flood peak estimation.

The remainder of this paper is structured as follows. Section 2 discusses the derivation of the first order partial derivative of the RBFN (and, briefly, the MLP for background) and includes a simple trigonometric example to show this working in practice. Section 3 introduces a real-world example and explains how rainfall-stage RBFN and MLP models were developed to modelling river levels in the River Ouse, UK. Section 4 presents the results of the sensitivity analysis using the partial derivative equations outlined in Section 2, before Section 5 concludes the paper with some thoughts and further work.

2. First Order Partial Derivatives of the RBFN and MLP

2.1. Introduction

In this section, we will focus on the calibration and the derivation of first order partial derivatives of the RBFN. The equivalent equation for the MLP is presented for completeness. A simple trigonometric function is presented by way of example to show the derived equations working in practice.

Both the MLP and RBFN are structured in a similar way—shown in **Figure 1**. This is a typical structure employed in hydrological modelling with a single hidden layer and a single output node. The n input nodes reside in Layer 0, the m hidden nodes in Layer 1, and the output node makes up Layer 2. The number of nodes in the input layer is determined by the data set (the number of predictors available) and then there is usually one output node (the predictand). The number of nodes in the hidden layer can be resolved by the training algorithm employed (for example, pruning algorithms for an MLP) but is often determined through a

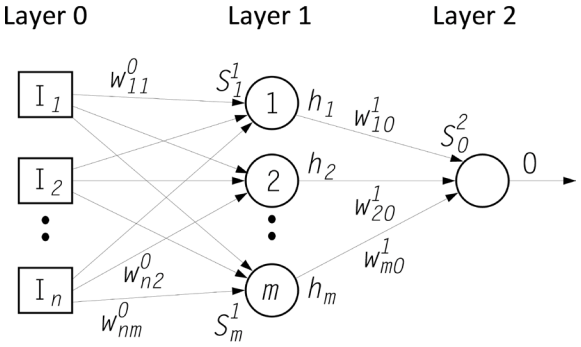

Figure 1. Typical feed forward neural network with one hidden layer and a single output unit (only some of the connection weights are shown for clarity).

trial and error approach. For an RBFN, the hidden nodes represent the basis functions employed.

2.2. RBFN

The RBFN consists of basis functions in the hidden layer and (typically) a linear activation function in the output layer. The basis functions in the hidden layer are typically represented by a Gaussian function (although it is acknowledged that other functions are sometimes used: [21]):

$$h(x) = e^{-(x-c)^2/2\sigma^2} \tag{1}$$

in which c represents the distance from the centre of the Gaussian function and σ represents its width (sphere of influence).

When calibrating an RBFN the first step involves establishing the centres of the m basis functions. If m is set to the number of training samples in the data set, the centres can be set to the values of the inputs of each training pair. However, for large data sets, this can lead to networks that are unwieldy and over-fitted. An alternative is to use some form of data-clustering such as k-means to identify the centres of a much smaller number of basis functions. These centres are represented by the connection weights between Layer 0 and Layer 1 in **Figure 1** (w_{ij}^0 for $i = 1$ to n inputs; $j = 1$ to m basis functions).

While the basis centres are calculated, the width of the basis functions, σ_j ($j = 1$ to m basis functions) are also determined. There are a number of ways this can be achieved, but the most common is to set the basis width to the root squared distance between the basis function and its next nearest neighbour.

With the basis functions established it remains for the weights connecting Layer 1 to Layer 2 to be calculated. With a linear activation function in the output layer the output O can be calculated as:

$$\boldsymbol{O} = \boldsymbol{w}H \tag{2}$$

in which $\boldsymbol{w}$ is the vector of weights connecting Layer 1 to Layer 2 and H is the arranged as:

$$H = \begin{bmatrix} h_1(X_1) & h_2(X_1) & \cdots & h_m(X_1) \\ \vdots & \vdots & & \vdots \\ h_1(X_p) & h_2(X_p) & \cdots & h_m(X_p) \end{bmatrix} \tag{3}$$

in which X_i represents the input vector of data sample i ($i = 1$ to p data samples).

Thus, the weights $\boldsymbol{w}$ can be calculated as:

$$\boldsymbol{w} = H^{-1}\boldsymbol{O} \tag{4}$$

Unfortunately, because H is not necessarily square, the pseudo inverse of H must be calculated:

$$\boldsymbol{w} = \left(H^{\mathrm{T}}H\right)^{-1} H^{\mathrm{T}}\boldsymbol{O} \tag{5}$$

With the RBFN calibrated the output from the network, O, can be calculated as follows. The output from each node (basis function) in the hidden layer, h_j $j =$

1 to m), is calculated from:

$$h_j = \mathrm{e}^{-S_j^1} \tag{6}$$

in which:

$$S_j^1 = \left(\left(I_1 - w_{1j}^0\right)^2 + \left(I_2 - w_{2j}^0\right)^2 + \cdots + \left(I_n - w_{nj}^0\right)^2\right) \Big/ 2\sigma_j^2 \tag{7}$$

The output from the network, O, can then be calculated as (w_{bO}^1 is the bias):

$$O = S_O^2 = h_1 w_{1O}^1 + h_2 w_{2O}^1 + h_3 w_{3O}^1 + \cdots + h_m w_{mO}^1 + w_{bO}^1 \tag{8}$$

2.3. RBFN Partial Derivative

The partial derivative of the output O with respect to input I_i is calculated as:

$$\frac{\partial O}{\partial I_i} = \sum_j \frac{\partial O}{\partial h_j} \frac{\partial h_j}{\partial S_j^1} \frac{\partial S_j^1}{\partial I_i} \tag{9}$$

for all hidden nodes, j.

In the case of the RBF (from (8)):

$$\frac{\partial O}{\partial h_j} = w_{jO}^1 \tag{10}$$

From (6):

$$\frac{\partial h_j}{\partial S_j^1} = -\mathrm{e}^{-S_j^1} = -h_j \tag{11}$$

From (7):

$$\frac{\partial S_j^1}{\partial I_i} = \left(I_i - w_{ij}^O\right) \Big/ \sigma_j^2 \tag{12}$$

Thus, by substitution, (9) becomes:

$$\frac{\partial O}{\partial I_i} = \sum_j \frac{w_{jO}^1 h_j \left(w_{ij}^0 - I_i\right)}{\sigma_j^2} \tag{13}$$

2.4. MLP Partial Derivative

An MLP is typically trained using the error back propagation algorithm (although many other training algorithms exist)—the detail of which is beyond the scope of this paper as it is covered in many other texts elsewhere. The connection weights between input i ($i = 1, \cdots, n$) and hidden node j ($j = 1, \cdots, m$) are denoted by w_{ij}^0 (0 representing layer 0); while the connection weights between hidden node j and the output node are donated by w_{jO}^1 (1 representing layer 1 and O representing the single output node). By convention, a bias is also added as an additional input to each node which enables the network to model more complex relationships. The biases are represented in as w_{bj}^0 for nodes in the hidden layer and w_{bO}^1 for the output node.

By using the backward chaining rule of Hashem [13] as before, for an MLP with sigmoid activation functions throughout, the partial derivative of the out-

put O with respect to input I_i is:

$$\frac{\partial O}{\partial I_i} = \sum_j O(1-O) w_{jO}^1 h_j (1-h_j) w_{ij}^0 \tag{14}$$

For $j = 1$ to m hidden nodes.

2.5. Trigonometric Example

As an example, take the simple trigonometric function as used by Hashem [13]:

$$y = \sin(4x) \tag{15}$$

Thus,

$$\frac{\mathrm{d}y}{\mathrm{d}x} = 4\cos(4x) \tag{16}$$

Data were generated for these two single-input, single-output functions for values between [−2, 2] in steps of 0.01. An MLP was trained with seven hidden units for 50,000 epochs and an RBFN was created with seven basis functions using these data. The first derivative for each model was then calculated using Equations 13 (for the RBFN—referred to as RBFN') and 14 (for the MLP—referred to as MLP').

Table 1 presents error measures (Root Mean Squared Error—RMSE; Coefficient of Determination Pearson R squared—RSqr) for both these models with respect to the full data set along with error measures for the calculated partial derivatives for both models (RBFN' and MLP'). Figure 2 shows the corresponding models (RBFN and MLP respectively) of Equation 15; and Figure 3 shows the RBFN' an MLP' models of the derivative given in Equation 16. RMSE and RSqr are standard error measures used to evaluate the performance of hydrological models [22]. They are calculated according to Equations 17 and 18: in which Q_i is the observed value ($i = 1$ to n values); $\hat{Q}_i$ is the modelled value ($i = 1$ to n values); $\bar{Q}$ is the mean of the observed values; and $\tilde{Q}$ is the mean of the modelled values.

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n} \left(\hat{Q}_i - Q_i\right)^2}{n}} \tag{17}$$

$$\mathrm{RSqr} = \left[\frac{\sum_{i=1}^{n} \left(Q_i - \bar{Q}\right)\left(\hat{Q} - \tilde{Q}\right)}{\sqrt{\sum_{i=1}^{n} \left(Q_i - \bar{Q}\right)^2 \sum_{i=1}^{n} \left(\hat{Q} - \tilde{Q}\right)^2}} \right]^2 \tag{18}$$

Table 1. Selected error measures for trigonometric models.

	RMSE	RSqr
RBFN	0.0495	0.9952
MLP	0.0452	0.9960
RBFN'	0.4575	0.9743
MLP'	0.5737	0.9573

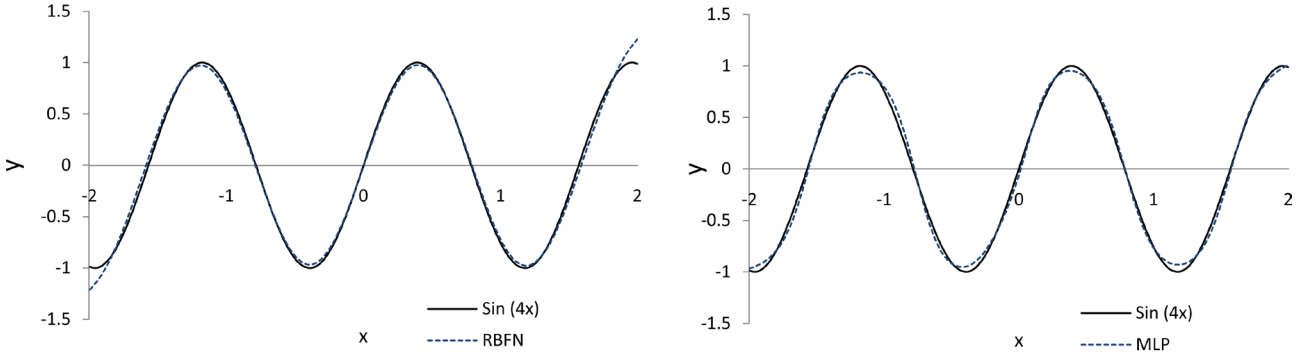

Figure 2. RBF and MLP models of $y = \sin(4x)$.

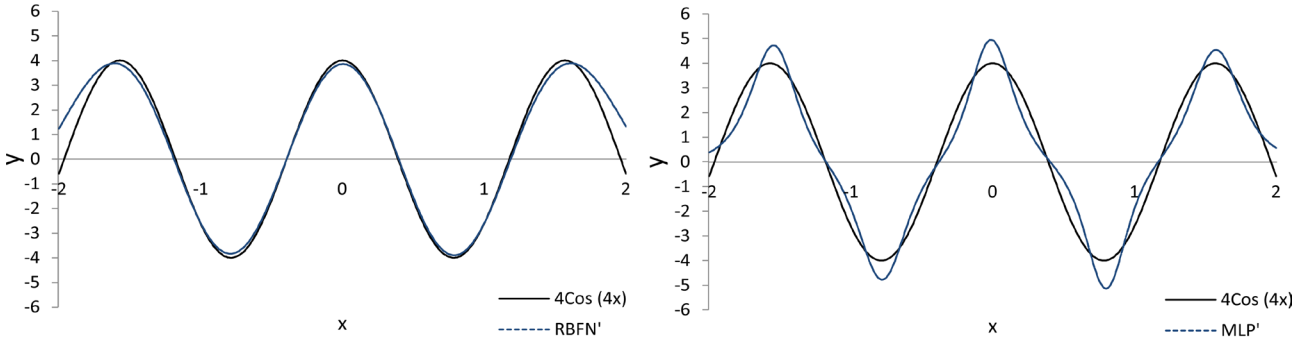

Figure 3. RBF and MLP sensitivity to input: equivalent of $y = 4\cos(4x)$.

Both models show "good" accuracy in modelling the underlying equation with RSqr values of over 99%. The first derivative also shows "good" accuracy with the RBFN' model, achieving an RSqr score of 97.4% while the MLP' achieves a score of 95.7%. Although the MLP' is marginally worse than the RBFN' according to both error measures, the results confirm the accuracy of the partial derivative equations presented in Equations (13) and (14).

3. Rainfall-Stage Forecasting

3.1. Introduction

Although the partial derivative equations presented for the RBF in the previous section are not new, their application within a hydrological context has yet to be explored. With this in mind, we present here a rainfall-stage model for the River Ouse in the UK. This has been used in previous studies such as [22] [23] [24].

3.2. Catchment

Six-hourly stage data for the River Ouse were available for this study. This catchment, situated in North Yorkshire, UK, covers an area of 3315 km^2 and contains an assorted mix of urban and rural land uses. It embraces three main rivers—the Swale, Ure and Nidd. It has a base flow index of 0.439; an average annual rainfall of 899 mm and its longest drainage path is 149.96 km. **Figure 4** provides an overview of this catchment and identifies the location of four river gauging stations (measuring stage in metres) and five rainfall stations (measur-

ing precipitation in millimetres) within the study area (the details of which are presented in Table 2 and Table 3).

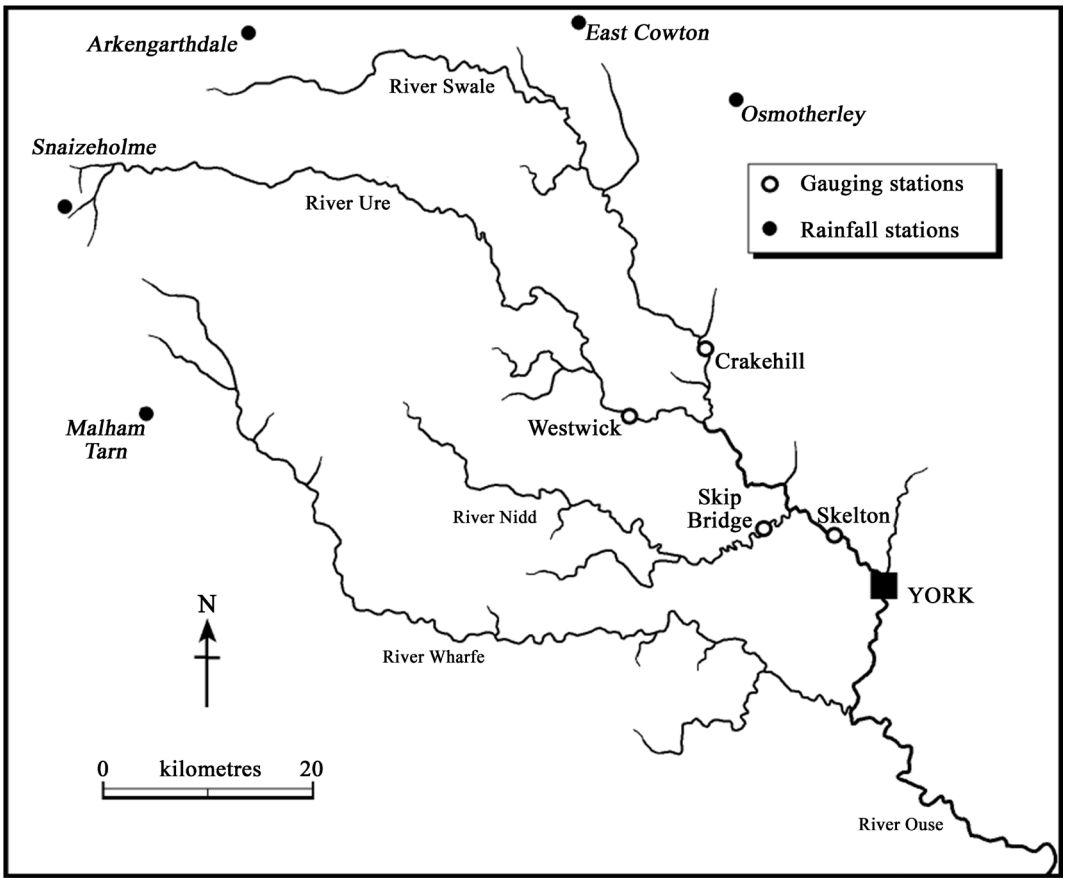

Figure 4. River Ouse catchment in North Yorkshire, UK.

Table 2. Gauging stations.

Name	River	Station Code	Grid Ref	Latitude	Longitude	Catchment Area (km²)
Crakehill	Swale	27071	SE 425733	54.153767	−1.3507459	1363
Skelton	Ouse	27009	SE 568553	53.990631	−1.1351792	3315
Skip Bridge	Nidd	27062	SE 482560	53.997793	−1.2662224	516
Westwick	Ure	27007	SE 355670	54.097678	−1.4586607	915

Table 3. Rainfall stations.

Name	Station Code	Grid Ref	Latitude	Longitude	Elevation (m)
Arkengarthdale	051684	NY 999030	54.4223	−2.00154	294
East Cowton	054261	NZ 308041	54.4313	−1.52516	48
Malham Tarn	073420	SD 893671	54.0996	−2.16228	381
Osmotherly	055223	SE 457967	54.3644	−1.29557	147
Snaizeholme	047282	SD 829866	54.2752	−2.26166	290

3.3. Modelling

In this study, we focus on modelling stage at the downstream site of Skelton ($Skelton_t$) using lagged water level (m) data at that location ($Skelton_{t-1}$); upstream stage at Crakehill ($Crakehill_{t-1}$); and a moving average (over 12 time steps) of rainfall (mm) at Tow Hill ($Tow\ Hill_{t-1}$). These three predictors were chosen as they allowed us to explore the sensitivity of the derived models with a lagged input, an upstream driver, and a rainfall component. Their selection is based on results from previous studies [23] and the strength of correlations with the predictand.

The data cover two winter periods between 1993 and 1996 (1 October to 31 March) and were split so that the winter of 1993-1994 was used for training (716 data points); while the winter of 1995-1996 was used for evaluation (720 data points). Note that in normal circumstances of ANN modelling three data sets are used—a training set, a validation set, and a test set. However, as the intention of these experiments was not to derive and test an independent model, but to explore the use of partial derivatives for sensitivity analysis, only two data sets were required. Thus, a training set was used to calibrate the models and the evaluation set was used to select the "best" models and perform the sensitivity analyses.

Several RBFNs were created using the training data set with 2, 4, 6, 8, 10, 12, 14, 16, 18, and 20 radial basis functions. Several MLPs were trained using error back propagation for 1000 to 20,000 epochs (in steps of 1000) with 2 - 10 hidden units. A learning rate of 0.1 and a momentum term of 0.9 were used. The RBFNs and MLPs were then evaluated (using RMSE) against the evaluation data set to choose the optimum configurations. The best RBFN had 10 basis functions; the best MLP had 9 hidden nodes and was trained for 2000 epochs.

Figure 5 and Figure 6 show the result of each model's output with respect to the validation period (RBFN and MLP respectively). Although the RBFN appears to model the flood peaks accurately, it seems to overestimate some of the low flow periods. Having said this, the error measures presented in Table 4 for the validation period provide reassurance of overall accuracy of the RBFN giving

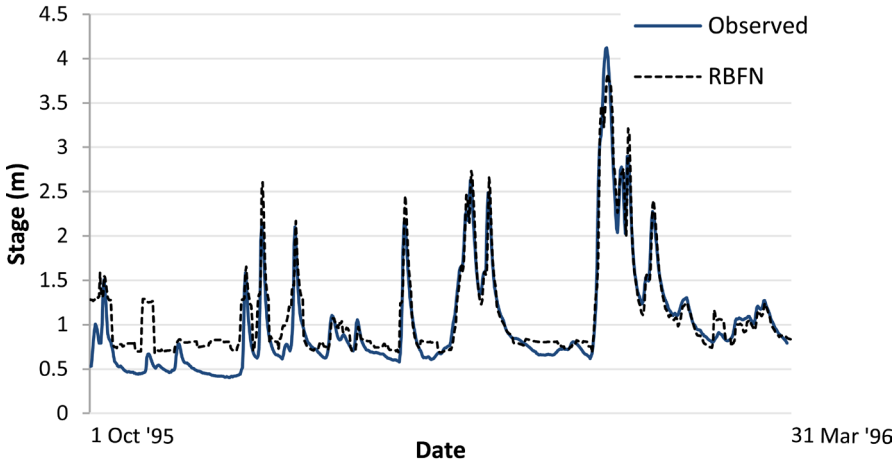

Figure 5. RBFN performance at modelling stage.

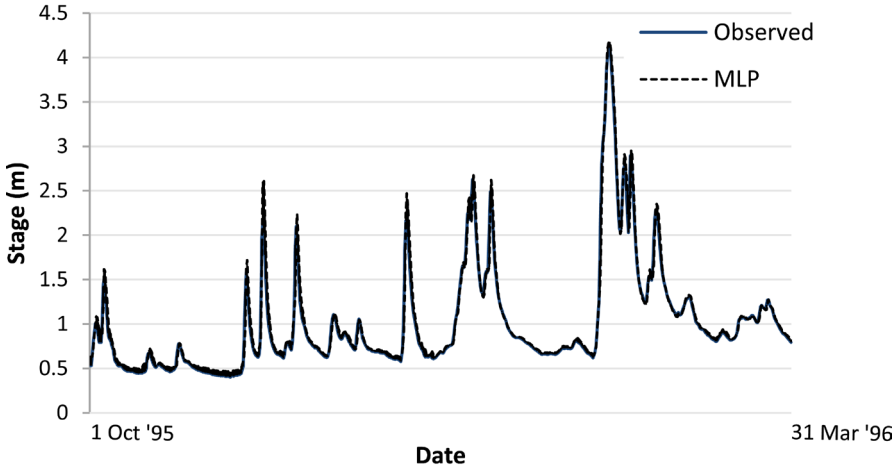

Figure 6. MLP performance at modelling stage.

Table 4. Selected error measures for the validation period.

	RMSE	RSqr
RBFN	0.2041	0.9149
MLP	0.0976	0.9772

an RSqr value of 91.49%.

4. Sensitivity Experiments

In this section, we apply Equations (13) and (14) to evaluate the sensitivity of the RBFN and MLP models to the three predictors. We present the results of the MLP first as these provide more meaningful insight than the RBFN which are presented later.

4.1. MLP Sensitivity Experiments

Note, because the data were standardised to [0.1, 0.9] for MLP training, the outputs from Equation 14 need adjusting by multiplying by range y/range x to return the actual dy/dx values. However, because the predictors have different ranges it would be wrong to compare dy/dx as changes in the predictor would not be consistent from one driver to the next. For example, if one predictor (say, x_1) ranges from 1 - 10; another (say, x_2) ranges from 1 - 1000, it would be wrong to compare change y/change x as it would take a large change in x_2 to affect y. However, in practice x_2 *would* be making large changes because we are dealing with real data. Therefore, Equation (14) is adjusted by multiplying by the range of each predictor to get the proportionate change in that predictor. This is a fairer comparison of the inputs.

Figures 7-9 show the sensitivity of the MLP model to each of the three inputs. As expected, the MLP is not particularly sensitive to rainfall (Tow Hill_{t-1})—exhibiting a gradually increasing level of sensitivity across the range of rainfall data. This is to be expected, as the catchment will not respond to low levels of

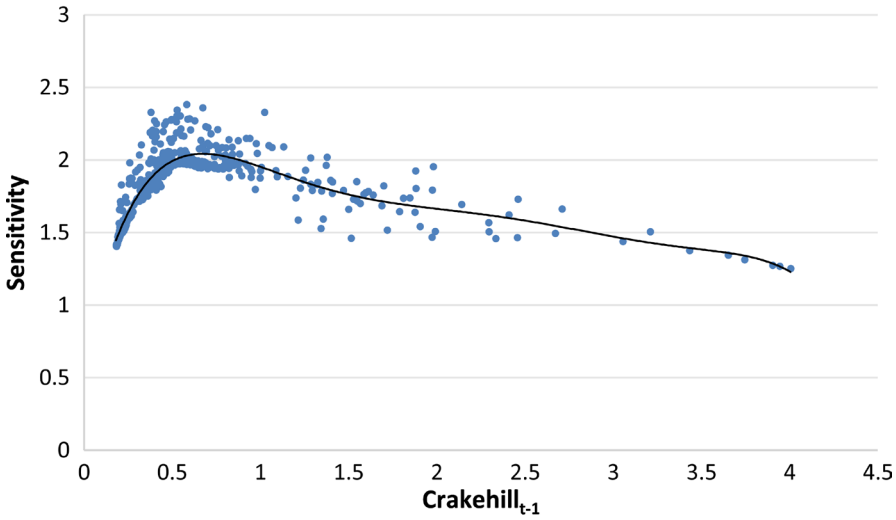

Figure 7. Sensitivity of MLP to Crakehill$_{t-1}$ predictor.

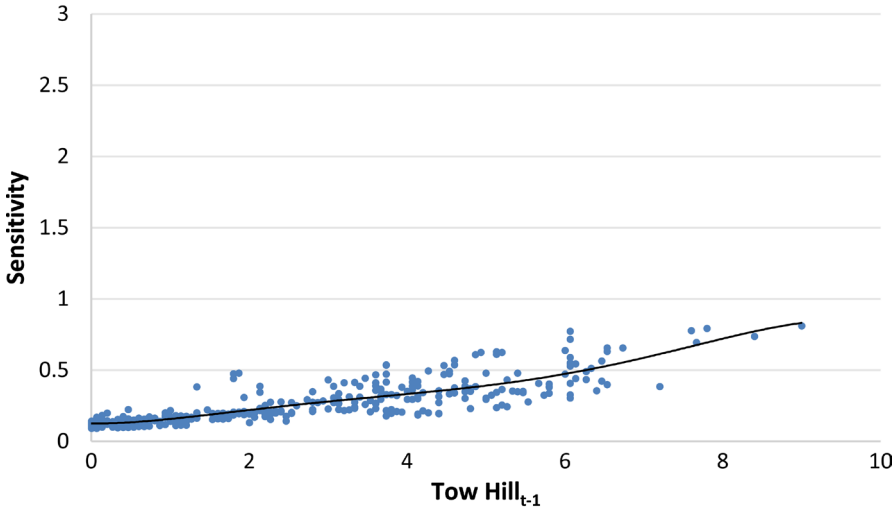

Figure 8. Sensitivity of MLP to Tow Hill$_{t-1}$ predictor.

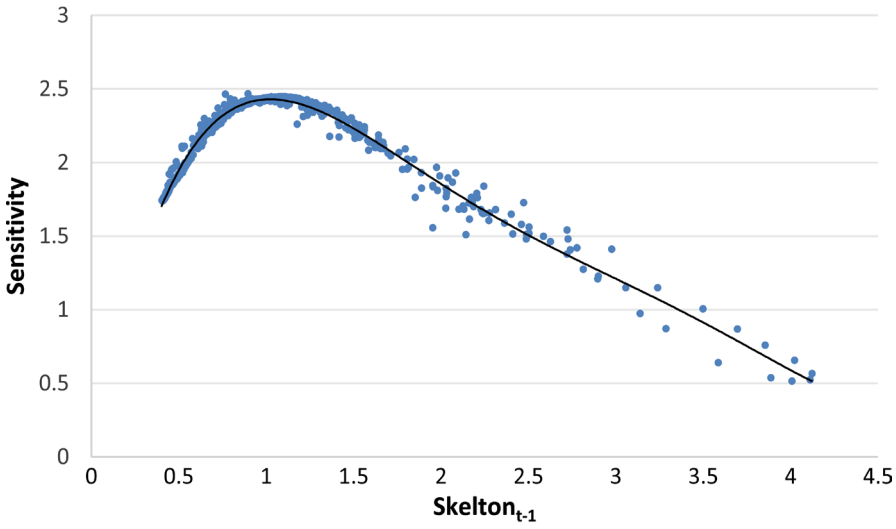

Figure 9. Sensitivity of MLP to Skelton$_{t-1}$ predictor.

rainfall which have little impact on base flow but will react to higher levels.

The MLP exhibits similar sensitivity levels to both upstream flow at Crakehill and antecedent flow at Skelton. What is interesting is how these sensitivities seem to tail off for—particularly for the Skelton$_{t-1}$ predictor. The answer to this may lie in the content of the training data set. Figure 10 and Figure 11 present frequency histograms of the data in the training set for the Skelton$_{t-1}$ and Crakehill$_{t-1}$ predictors, respectively. The available data for Skelton$_{t-1}$ steadily decreases for higher values. This broadly follows the sensitivity of this driver. With fewer data points available at higher flows, the MLP has not been able to capture the behaviour of the catchment as well as it was at lower flows, where data were more abundant and more differentiation between values could be made.

A similar scenario has occurred for the Crakehill$_{t-1}$ predictor, but to a lesser extent. In this case, above values of around 1 m, the frequency of data does not tail off as much as the data did for Skelton$_{t-1}$. This has probably led to a less severe decline in the sensitivity of the model to Crakehill$_{t-1}$ which decreases only

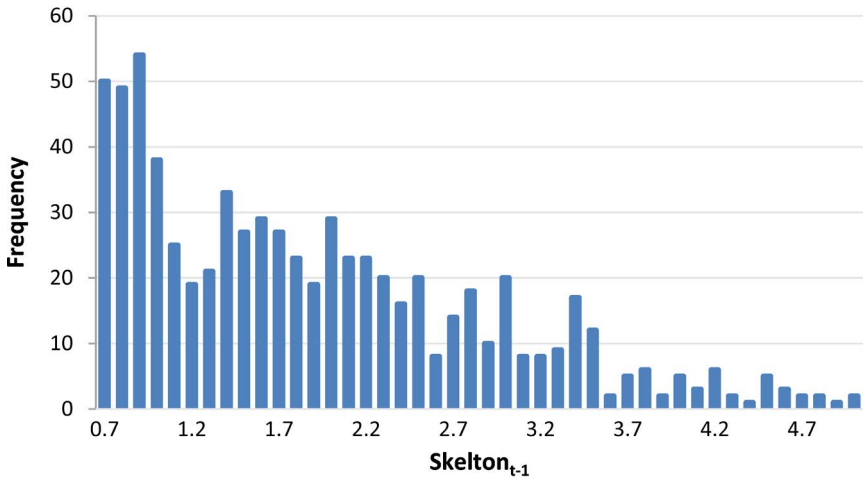

Figure 10. Frequency distribution of Skelton$_{t-1}$ data in the validation data set.

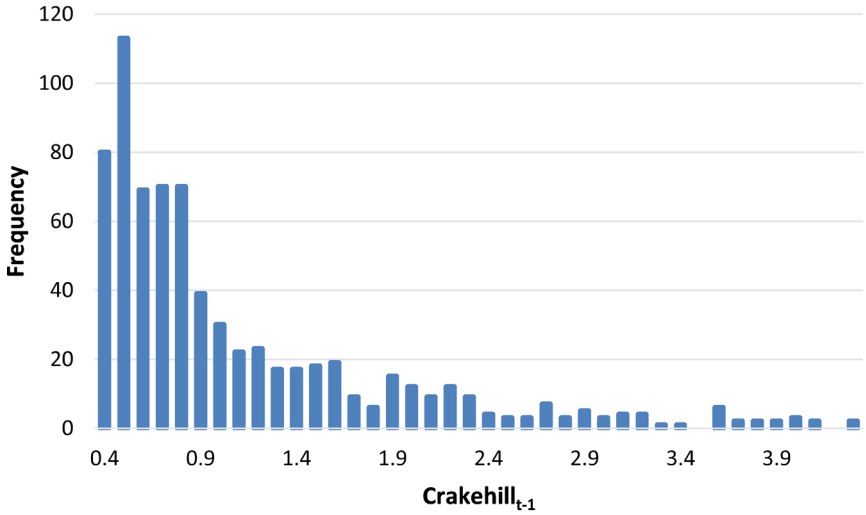

Figure 11. Frequency distribution of Crakehill$_{t-1}$ data in the validation data set.

slightly for values over 1 m.

In conclusion, Equation (13) does provide a way of exploring the sensitivity of a real-world model to different inputs. Investigations of the sensitivities give insight into the model behaviour and provide the user with a better understanding of why the model performs as it does and how data can affect the training and performance of the final model.

4.2. RBFN Sensitivity Experiments

Figures 12-14 show the sensitivity of the RBFN (10 basis functions) to each of the three inputs which are calculated using Equation 13. A best fit polynomial line of order three has been added to these plots to highlight the general shape of the sensitivity. The sensitivity of the model to Crakehill_{t-1} and Skelton_{t-1} are similar in nature—rising to a peak before falling away again with similar sensitivity values. The sensitivity of the model to Tow Hill_{t-1} is less clear appearing to rise and fall across the range of values. The sensitivities of Crakehill_{t-1} and

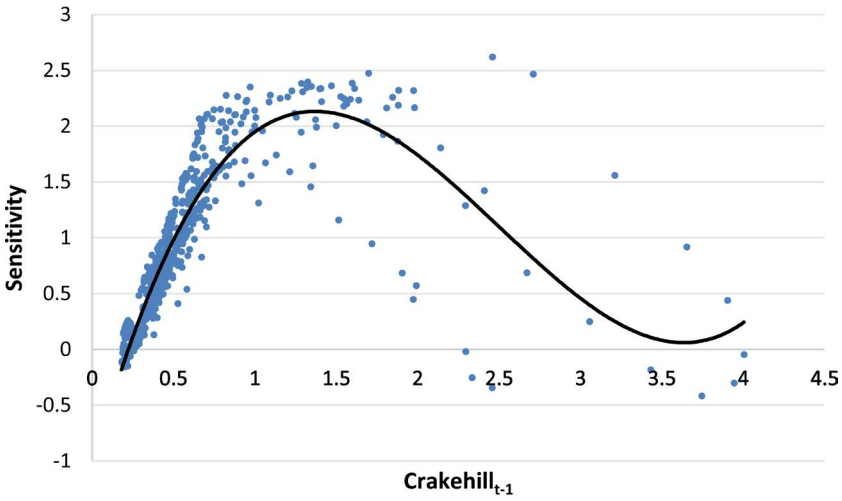

Figure 12. Sensitivity of RBFN to Crakehill_{t-1} predictor.

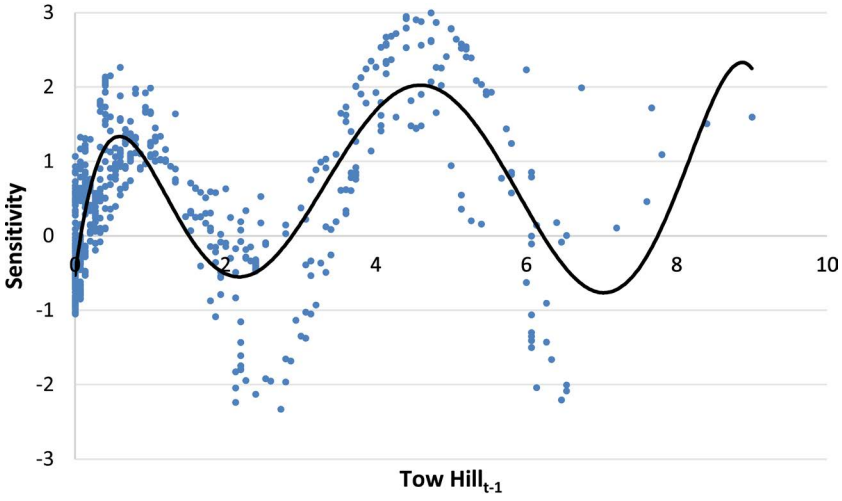

Figure 13. Sensitivity of RBFN to Tow Hill_{t-1} predictor.

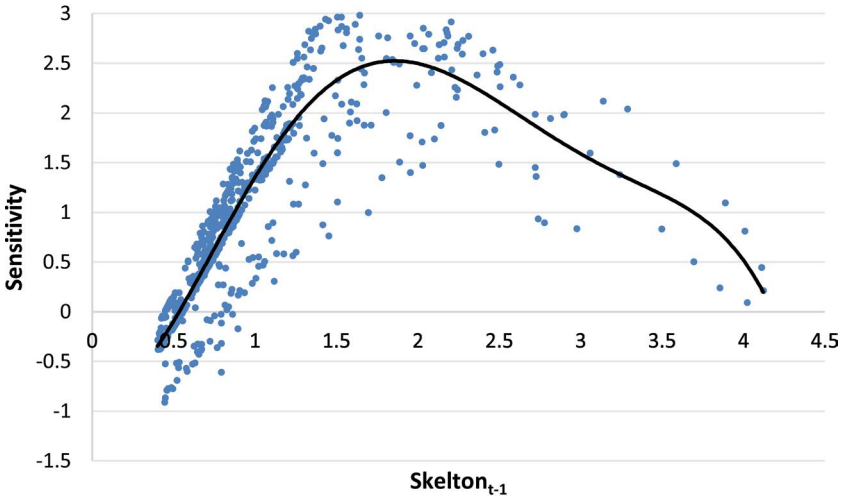

Figure 14. Sensitivity of RBFN to Skelton$_{t-1}$ predictor.

Skelton$_{t-1}$ do not really provide any meaningful physical interpretation of how the model is behaving—for example, why would the model have low sensitivity for small Crakehill$_{t-1}$ values, high sensitivity for values of around 1.5 m, and low sensitivity again for values of Crakehill$_{t-1}$ above around 3 m? Equally puzzling are the fluctuating sensitivity values of the model with respect to the rainfall input of Tow Hill. The conclusion would be that these sensitivities, rather than representing a physical relationship within the RBFN model, are presenting the behaviour of the RBFN itself in which the model is distributed throughout the basis centres.

As these sensitivities may be peculiar to the RBFN chosen (*i.e.* the one with 10 basis functions) the sensitivities of all other RBFNs (from 2 to 20 basis functions in steps of 2) were explored for each of the three inputs. These sensitivities are presented in Figures 15-17 (for Skelton$_{t-1}$, Tow Hill$_{t-1}$ and Crakehill$_{t-1}$ respectively). Although no obvious physical pattern emerges, the figures do show how the models become more pronounced as more basis centres are used. The conclusion in this case is that the models are reflecting the physical relationship in the catchment as a distributed relationship with the basis functions themselves.

The lack of any physical meaning in the interpretation of the sensitivities of the RBFN models to their inputs led to an additional question of whether they were representing the physical nature of the model within the basis functions themselves. With this idea in mind, the RBFNs were analysed further by determining which of their basis functions was aligned most closely to each individual data point. In other words, for each data point, we identified (by Euclidian distance) which was its nearest basis function centre in each model. Figure 18 and Figure 19 show these results for the RBFNs with two and three basis functions, respectively. These figures show by colour, for each data point in the data set, which of the two (or three) basis centres the model has aligned the data point with. Only the two and three basis function models are presented as more clusters over-complicates the plots and makes relationships less clearly visible.

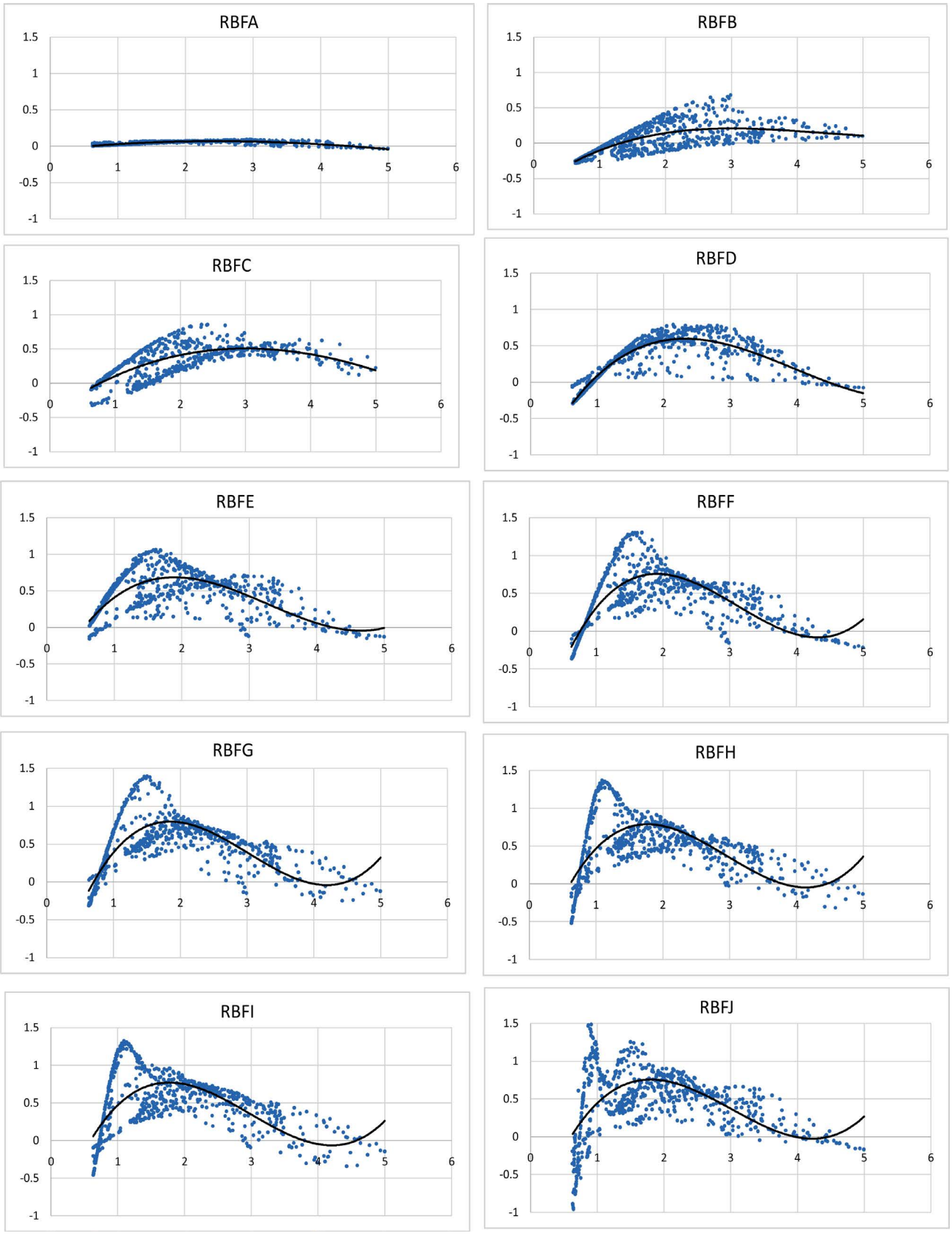

Figure 15. Sensitivity of RBF to Skelton$_{t-1}$ values. A to J represent increasing numbers of basis functions from two to twenty (steps of two).

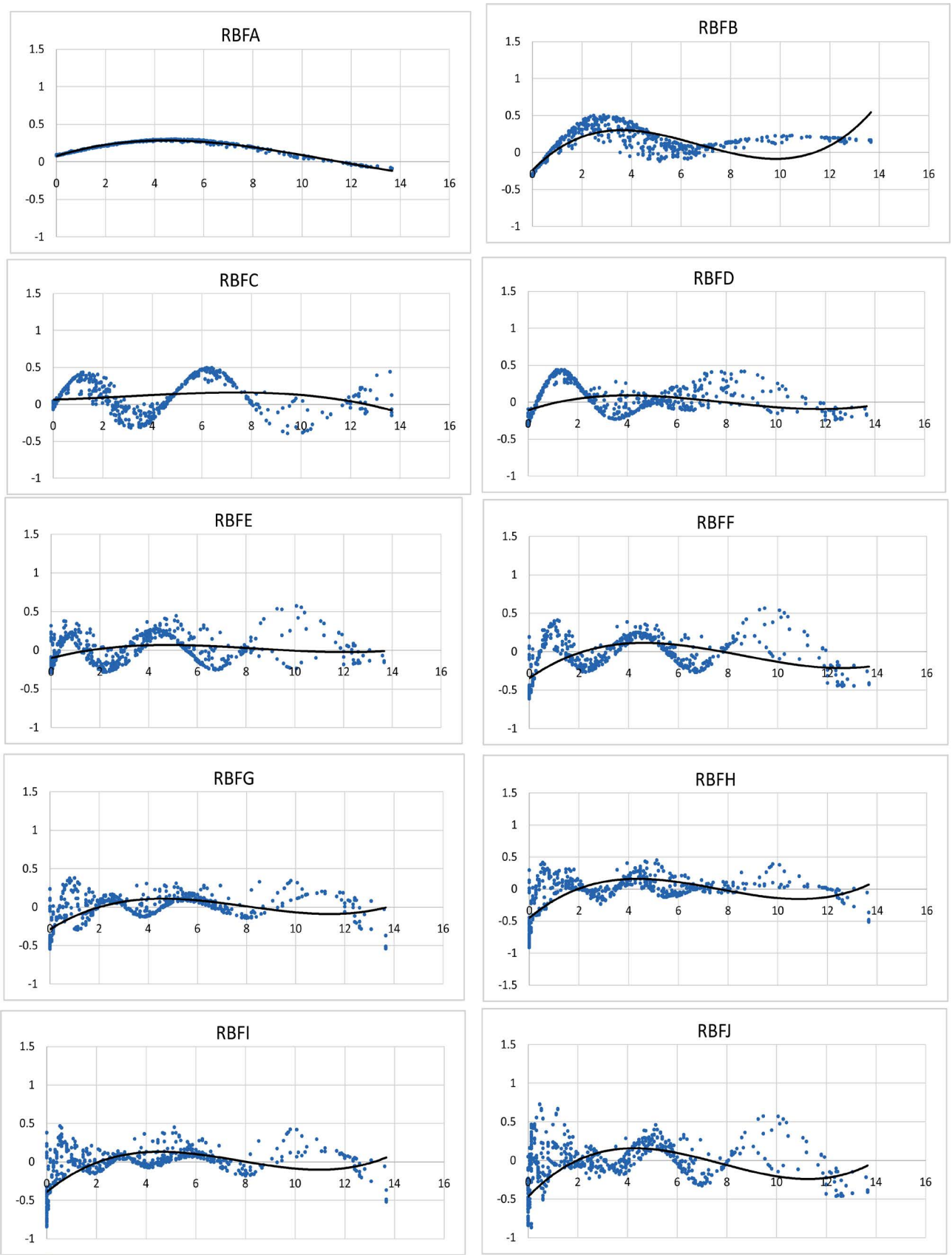

Figure 16. Sensitivity of RBF to Tow Hill_{t-1} values.

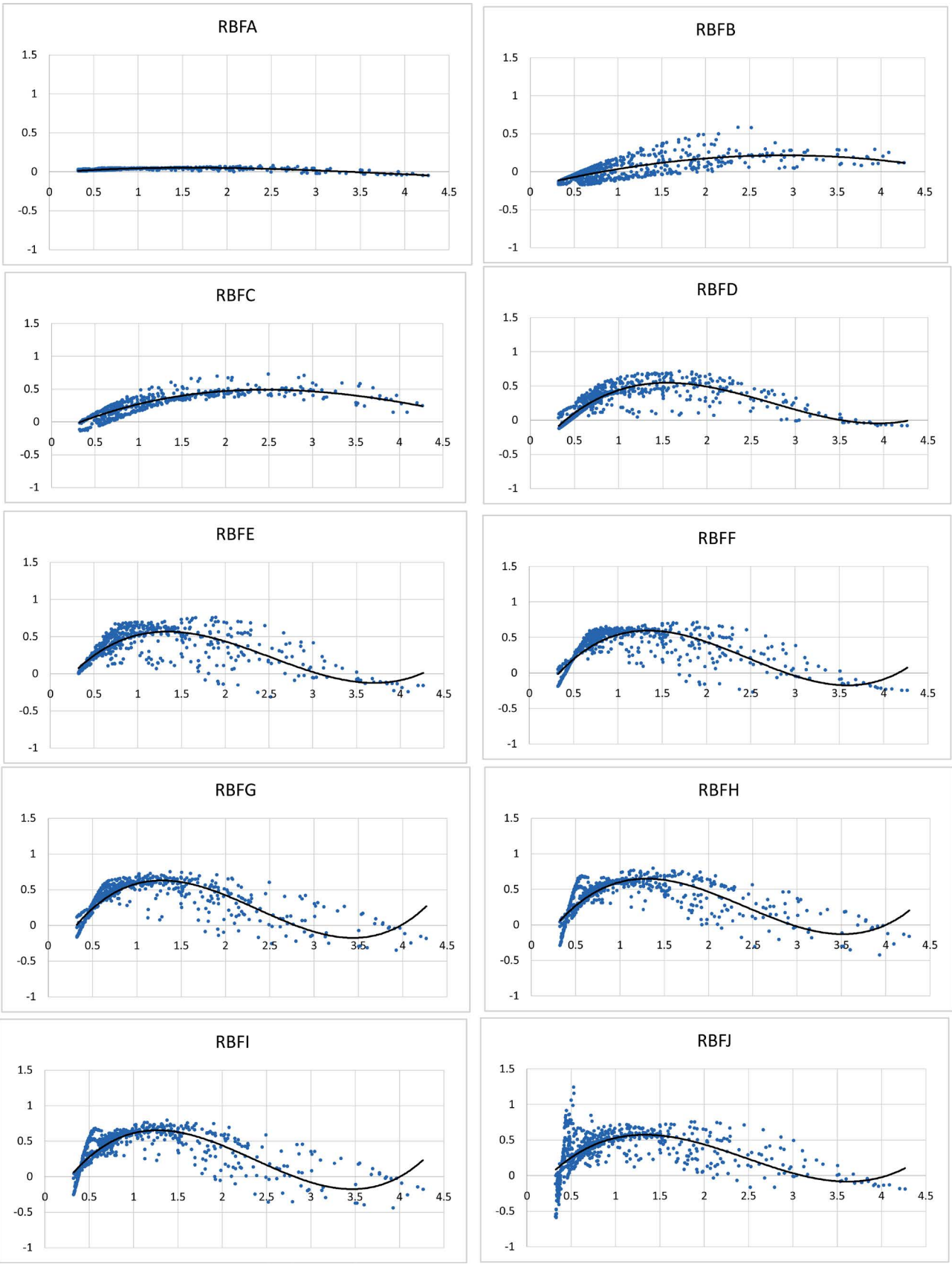

Figure 17. Sensitivity of RBF to Crakehill$_{t-1}$ values.

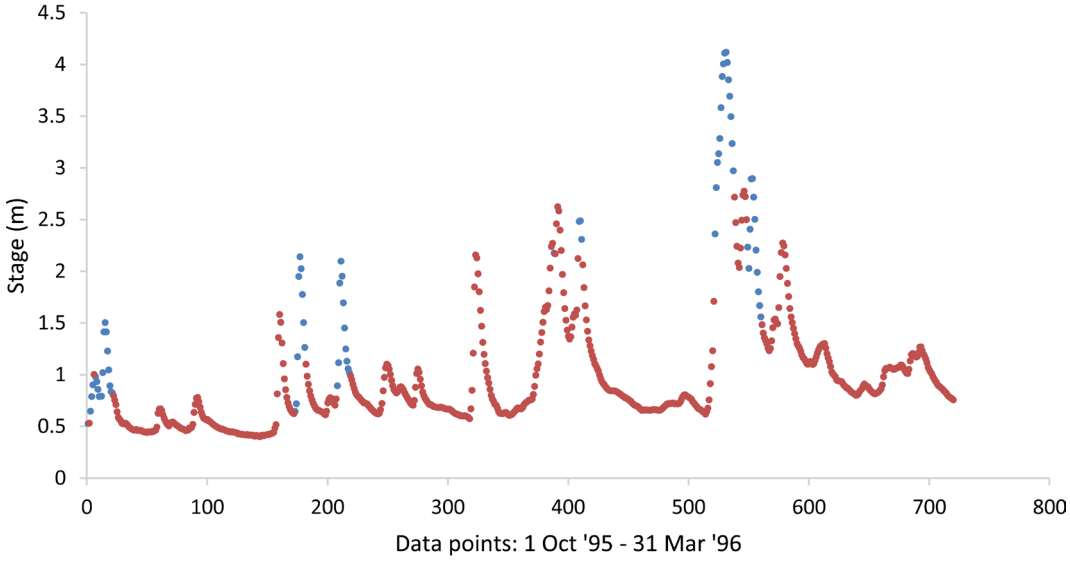

Figure 18. Data aligned with nearest RBF (2 clusters).

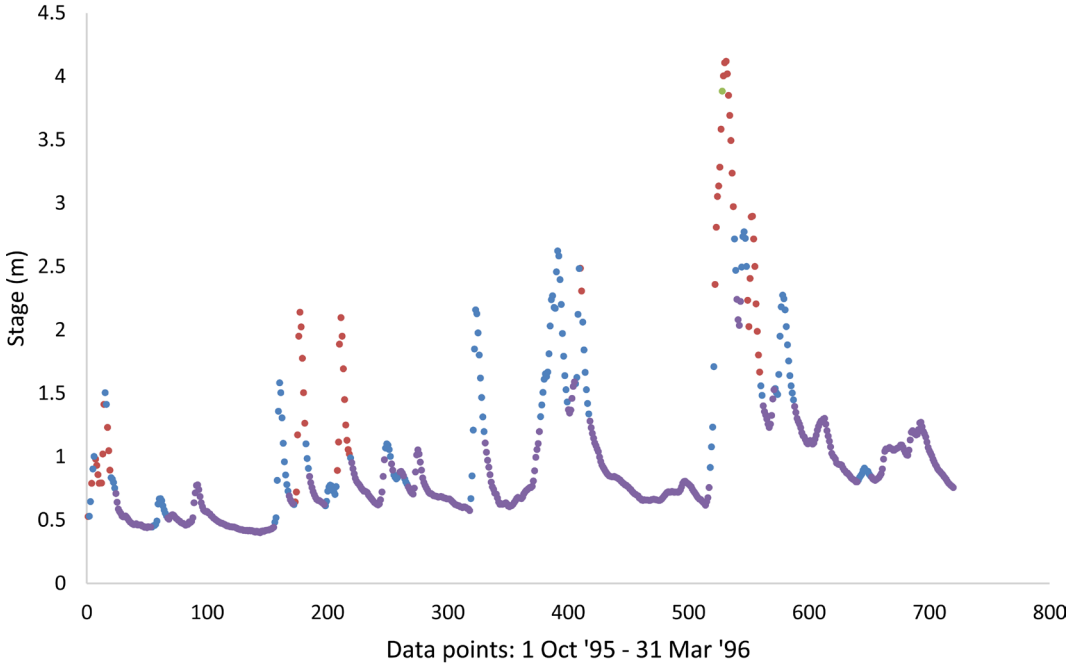

Figure 19. Data aligned with nearest RBF (3 clusters).

Figure 18 (two basis functions) appears to show that one basis function is picking out peak stage (blue) while the other is modelling the low stage component (red). A similar result is seen in Figure 19 in which one basis function is clearly aligned with low stage events (purple), one with peak stages (red) and one with intermediate stage (blue)—particularly within a given time frame. These results show that the model is picking out some physical relationship within the data that was not clear from the sensitivity analysis performed earlier. Although the RBFN is behaving as a black box in terms of the relationship between its inputs and its outputs, there is an underlying physical behaviour within

the model that can be explored by reviewing the basis functions and their links with the data.

5. Conclusions

This paper has applied, for the first time, a new means of directly calculating the partial derivatives of inputs with respect to the output in RBFNs for a rainfall-stage model. The derived solution is shown to work in simple cases and a complex, real world example for the MLP. Such analysis is particularly important as it allows modellers to explore the behaviour of their models more thoroughly, justify their performance and determine their ability to generalise. While such an analysis appears to work for an MLP, the results are less conclusive in the case of RBFNs. This leads us to the conclusion that such a technique does not work well real-world applications of RBFNs and alternative analysis should be performed. Such an analysis can involve exploring the relationship between each of the basis functions with the data points themselves. A preliminary analysis with the data set used in this study shows promise—in that there seems to be a clear relationship between aspects of the hydrograph and individual basis functions.

The results presented in this paper lead to several further investigations. First, it would be prudent to determine partial derivatives for RBFs that use alternative basis functions to the popular Gaussian function as these may provide more meaningful physical rationality. Second, looking at how the information garnered from such analysis can help to explain, justify, evaluate, and generalise from such models. There is currently no framework whereby sensitivity results can be used to derive such information. Such a framework would prove beneficial in the evaluation of neural networks and go a long way to addressing the criticisms levelled at such models due to their black-box behaviour. Finally, exploring the relationship between basis functions and components of the hydrograph shows promise from this work.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Ghorbani, M.A., Zadeh, H.A., Isazadeh, M. and Terzi, O. (2016) A Comparative Study of Artificial Neural Networks (MLP, RBF) and Support Vector Machine Models for River Flow Prediction. *Environmental Earth Sciences*, **75**, 1-14. https://doi.org/10.1007/s12665-015-5096-x

[2] Nie, S., Bian, J, Wan, H., Sun, X. and Zhang, B. (2017) Simulation and Uncertainty Analysis for Groundwater Levels Using Radial Basis Function Neural Network and Support Vector Machine Models. *Journal of Water Supply: Research and Technology*, **66**, 15-24. https://doi.org/10.2166/aqua.2016.069

[3] Nawaz, N., Harun, S., Othman, R. and Heryansyah, A. (2016) Application of Radial

Basis Function Neural Networks for Modelling Rainfall-Runoff Processes: A Case Study of Semenyih River Catchment, Malaysia. *Chiang Mai Journal of Science*, **43**, 1358-1367.

[4] Chai, S.S., Wong, W.K. and Goh, K.L. (2017) Rainfall Classification for Flood Prediction Using Meteorology Data of Kuching, Sarawak, Malaysia: Backpropagation vs Radial Basis Function Neural Network. *International Journal of Environmental Science and Development*, **8**, 385-388. https://doi.org/10.18178/ijesd.2017.8.5.982

[5] Broomhead, D.S. and Lowe, D. (1988) Multivariable Function Interpolation and Adaptive Networks. *Complex Systems*, **2**, 321-355.

[6] Moody, J.E. and Darkin, C.J. (1989) Fast Learning in Networks of Locally-Tuned Processing Units. *Neural Computation*, **1**, 281-294. https://doi.org/10.1162/neco.1989.1.2.281

[7] Park, J. and Sandberg, I.W. (1993) Approximation and Radial Basis Function Networks. *Neural Computation*, **5**, 305-316. https://doi.org/10.1162/neco.1993.5.2.305

[8] Powell, M.J.D. (1987) Radial Basis Functions for Multivariable Interpolation: A Review. In: Mason, J.C. and Cox, M.G., Eds., *Algorithms for Approximation*, Clarendon Press, Oxford, 143-167.

[9] Chen, X., Zeng, X., Chu, R. and Zhong, S. (2010) A Quantified Sensitivity Measure of Radial Basis Function Neural Networks to Input Variation. *IJCNN*, 2010 *Joint Conference on Neural Networks*, Barcelona, 18-23 July 2010, 1-6. https://doi.org/10.1109/IJCNN.2010.5596949

[10] Stevenson, M., Winter, R. and Widrow, B. (1990) Sensitivity of Feed forward Neural Networks to Weight Errors. *IEEE Transactions on Neural Networks*, **1**, 71-80. https://doi.org/10.1109/72.80206

[11] Piche, S.W. (1995) The Selection of Weight Accuracies for Madalines. *IEEE Transactions on Neural Networks*, **6**, 432-445. https://doi.org/10.1109/72.363478

[12] Zeng, X., Wang, Y. and Zhang, K. (2006) Computation of Adaline's Sensitivity to Weight Perturbations. *IEEE Transactions on Neural Networks*, **17**, 515-519. https://doi.org/10.1109/TNN.2005.863418

[13] Hashem, S. (1992) Sensitivity Analysis for Feed forward Artificial Neural Networks with Differentiable Activation Functions. *International Joint Conference on Neural Networks* 1992, Vol. 1, Baltimore, 419-424.

[14] Yang, J., Zeng, X.Q. and Zhong, S.M. (2013) Computation of Multilayer Perceptron Sensitivity to Input Perturbation. *Neurocomputing*, **99**, 390-398. https://doi.org/10.1016/j.neucom.2012.07.020

[15] Zeng, X.Q. and Yeung, D.S. (2001) Sensitivity Analysis of Multilayer Perceptron to Input and Weight Perturbations. *IEEE Transactions on Neural Networks*, **12**, 1358-1366. https://doi.org/10.1109/72.963772

[16] Huang, L.H., Zeng, X.Q., Zhong, S.M. and Han, L.X. (2014) Sensitivity Study of Binary Feed forward Neural Networks. *Neurocomputing*, **136**, 268-280. https://doi.org/10.1016/j.neucom.2014.01.005

[17] Ng, W.W.Y., Yeung, D.S., Ran, Q. and Tsang, E.C.C. (2002) Statistical Output Sensitivity to Input and Weight Perturbations of Radial Basis Function Neural Networks. *IEEE IC-SMC*, Tunisia, 6-9 October 2002, 503-508.

[18] Shi, D., Yeung, D.S. and Gao, J. (2005) Sensitivity Analysis Applied to the Construction of Radial Basis Function Networks. *IEEE Transactions on Neural Networks*, **18**, 951-957. https://doi.org/10.1016/j.neunet.2005.02.006

[19] Wang, X., Li, C., Yeung, D.S., Song, S. and Feng, H. (2008) A Definition of Partial

Derivative of Random Functions and Its Application to RBFNN Sensitivity Analysis. *Neurocomputing*, **71**, 1515-1526. https://doi.org/10.1016/j.neucom.2007.05.005

[20] Wing, W.Y., Ng, Z.H., Yeung, D.S. and Chan, P.K. (2014) Steganalysis Classifier Training via Minimizing Sensitivity for Different Imaging Sources. *Information Sciences*, **281**, 211-224. https://doi.org/10.1016/j.ins.2014.05.028

[21] Harpham, C. and Dawson, C.W. (2006) The Effect of Different Basis Functions on a Radial Basis Function Network for Time Series Prediction: A Comparative Study. *Neurocomputing*, **69**, 2161-2170. https://doi.org/10.1016/j.neucom.2005.07.010

[22] Dawson, C.W., Abrahart, R.J. and See, L.M. (2007) HydroTest: A Web-Based Toolbox of Evaluation Metrics for the Standardised Assessment of Hydrological Forecasts. *Environmental Modelling and Software*, **22**, 1034-1052. https://doi.org/10.1016/j.envsoft.2006.06.008

[23] Dawson, C.W., See, L.M., Abrahart, R.J. and Heppenstall, A.J. (2006) Symbiotic Adaptive Neuro-Evolution Applied to Rainfall-Runoff Modelling in Northern England. *Neural Networks*, **19**, 236-247. https://doi.org/10.1016/j.neunet.2006.01.009

[24] Dawson, C.W., Abrahart, R.J. and See, L.M. (2010) HydroTest: Further Development of a Web Resource for the Standardised Assessment of Hydrological Models. *Environmental Modelling & Software*, **25**, 1481-1482. https://doi.org/10.1016/j.envsoft.2009.01.001