

EMRBF: A Statistical Basis for Using Radial Basis Functions for Process Control

Lyle H. Ungar
Department of Chemical Engineering
University of Pennsylvania
ungar@cis.upenn.edu

Richard D. De Veaux
Mathematics Department
Williams College
deveaux@williams.edu

Abstract

Radial Basis Function (RBF) neural networks offer an attractive equation form for use in model-based control because they can approximate highly nonlinear plants and yet are well suited for linear adaptive control. We show how interpreting RBFs as mixtures of Gaussians allows the application of many statistical tools including the EM algorithm for parameter estimation. The resulting EMRBF models give uncertainty estimates and warn when they are extrapolating beyond the region where training data was available.

1. Introduction

Radial basis functions (RBFs), are a form of neural network for approximating nonlinear relationships. They typically take the form

$$\mathbf{y} = \sum_{j=1}^k \mathbf{w}_j \phi_j(\mathbf{x}) \quad (1)$$

where the functions $\phi_j(\mathbf{x})$ are Gaussian basis functions with centers μ_j and widths σ_j or, as we will interpret them in this paper, Gaussian density functions with mean μ_j and standard deviation σ_j .

A number of different methods have been used to pick the centers and widths of the basis functions. The simplest is to pick a fixed set of basis functions, for example with evenly spaced centers. However, when one is modeling a system with multiple inputs, picking basis functions adaptively, i.e. based on the data at hand, gives much more accurate models with less data than using fixed basis functions [1].

In control applications, the mapping that the RBFs represent can be either a model of a plant (where e.g. \mathbf{x} is the current state of the plant and the current

control action and \mathbf{y} is the next state of the plant) or a controller (where e.g. \mathbf{x} is the current state of the plant and the desired next state of the plant and \mathbf{y} is the control action to be taken). In real applications, the true state is generally not measurable, so an ARMA-style input consisting of a window of current and past measurements is used. Once one has a plant model, it can be incorporated into any model-based nonlinear control scheme such as MPC [11,4]

Radial basis functions share many of the advantages of conventional feed forward neural networks (“backpropagation networks”). In the limit, both types of networks, with enough neurons and enough data, can approximate any well-behaved function arbitrarily well [10]. In practice, RBFs are often superior to backpropagation networks when the input is of relatively low dimension (5-10 inputs), but are usually inferior for high dimensional problems (over 20 inputs).

RBFs offer a number of advantages over backpropagation networks. As we will show below, RBFs lend themselves to producing error bars on their predictions. They can easily be used to learn model mismatch when an approximate first principles model is available [5]. And, most attractively for adaptive control purposes, if one fixes the basis functions (i.e. picks the centers and widths of Gaussians), then the predictions are linear in the coefficients (the weights). These weights can then be adapted, and all of the standard mathematics of linear systems can be applied. There is a large literature on the use of RBFs for control applications; see, for example, Parthasarathy and Narendra [9] and Sanner and Slotine [12].

We present a statistical interpretation of RBFs as a mixture of Gaussians, and show how this leads to an efficient algorithm for simultaneously estimated the centers and widths, μ_j and σ_j , of the basis functions and the coefficients, \mathbf{w}_j . This method, called EMRBF (Expectation Maximization Radial Basis Functions), picks basis functions taking into account both

the \mathbf{x} 's and the \mathbf{y} 's unlike the standard k-means clustering on the x 's [8]. We then present results comparing EMRBF with standard RBF estimation methods. The final section summarizes and mentions additional benefits from using EMRBF for control.

2. Mixture Model Formulation

One way of describing radial basis functions is to view them as mixtures of Gaussians. This allows a statistical interpretation which is less ad hoc than standard RBFs and allows the application of the EM algorithm. Assume that one has a set of n data points, \mathbf{z}_i , each drawn from one of k different populations, with probability λ_j , where the subscript j denotes the population. Further assume that for each population j , the \mathbf{z}_i are distributed as multivariate Gaussians with mean $\boldsymbol{\nu}_j$ and covariance $\boldsymbol{\Sigma}_j$. Under specific assumptions about the form of the covariance matrix this gives a radial basis function.

One can consider the points \mathbf{z}_i as being composed of two parts, \mathbf{x}_i and \mathbf{y}_i , $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$, where \mathbf{x} represents inputs to a function $\mathbf{y} = f(\mathbf{x})$. If one takes $\boldsymbol{\nu}_j = (\boldsymbol{\mu}_j, \mathbf{w}_j)$, i.e. the means of the \mathbf{x} 's and \mathbf{y} 's produced by the j -th population are $\boldsymbol{\mu}_j$ and \mathbf{w}_j , respectively, and assumes that the \mathbf{x} 's and \mathbf{y} 's are uncorrelated and have variance σ_j^2 and ϕ_j^2 , then one can calculate the expected value of the \mathbf{y} corresponding to a new \mathbf{x} by

$$\hat{\mathbf{y}} = \sum_{j=1}^k \mathbf{w}_j P_j(\mathbf{x}) \quad (2)$$

where

$$P_j(\mathbf{x}) = \frac{\frac{\lambda_j}{\sigma_j} \exp\left\{-\frac{\|\mathbf{x}-\boldsymbol{\mu}_j\|^2}{2\sigma_j^2}\right\}}{\sum_{t=1}^k \frac{\lambda_t}{\sigma_t} \exp\left\{-\frac{\|\mathbf{x}-\boldsymbol{\mu}_t\|^2}{2\sigma_t^2}\right\}} \quad (3)$$

This is, of course, equivalent to a radial basis function in which the basis functions have been normalized. The weights, \mathbf{w}_j , are the expected values of the \mathbf{y} 's and the centers and widths of the radial basis functions are the means and the standard deviations of the \mathbf{x} 's.

We can use this statistical interpretation to derive an EM algorithm for simultaneously calculating the basis functions as the weights. As above, assume k populations and n data points $\mathbf{z}_i = (\mathbf{x}_i, \mathbf{y}_i)$ where each

\mathbf{z}_i is an element of population j with probability τ_{ij} . The probability density function for \mathbf{z}_i is taken to be given by the Normal distribution $f_j(\mathbf{z}_i) = N(\boldsymbol{\nu}_j, \boldsymbol{\Sigma}_j)$ where $\boldsymbol{\nu}_j$ is defined as above to be $(\boldsymbol{\mu}_j, \mathbf{w}_j)$.

The overall probability density function for \mathbf{z}_i is then given by the weighted sum:

$$f(\mathbf{z}_i) = \sum_{j=1}^k \lambda_j f_j(\mathbf{z}_i) \quad j = 1, \dots, k \quad (4)$$

where λ_j is the probability than an unknown point is from population j (i.e. λ_j gives the relative population sizes).

Several different forms of the covariance matrix can be considered. A simple model, which generates the standard RBF, is to assume that all elements of \mathbf{x} and \mathbf{y} are uncorrelated and that within a population j , all elements of \mathbf{x} have variance σ_j^2 and all elements of \mathbf{y} have variance ϕ_j^2 . In this case, $f_j(\mathbf{z}_i) =$

$$\frac{1}{(2\pi)^{(l+m)/2} \sigma_j^l \phi_j^m} \exp\left\{-\frac{\|\mathbf{x}_i - \boldsymbol{\mu}_j\|^2}{2\sigma_j^2} - \frac{\|\mathbf{y}_i - \mathbf{w}_j\|^2}{2\phi_j^2}\right\} \quad (5)$$

where l is the dimension of \mathbf{x} and m is the dimension of \mathbf{y} . One can equally easily assume that the \mathbf{x} 's are correlated with themselves and the \mathbf{y} 's have correlation as well. This gives a covariance with a block matrix form and produces elliptical basis functions [13]. One can also use a full covariance matrix.

The following results then follow from the mixture models literature (See e.g. MacLachlan and Basford, 1988, p. 37), which we have specialized for the case where $\boldsymbol{\Sigma}_j$ takes the diagonal form described above.

The population densities are:

$$\hat{\lambda}_j = \sum_{i=1}^n \hat{\tau}_{ij} / n \quad j = 1, \dots, k \quad (6)$$

The means and variances of the different populations (i.e. the centers and widths of the basis functions are:

$$\hat{\boldsymbol{\mu}}_j = \sum_{i=1}^n \hat{\tau}_{ij} \mathbf{x}_i / n \hat{\lambda}_j \quad (7)$$

and

$$\hat{\sigma}_j^2 = \sum_{i=1}^n \hat{\tau}_{ij} (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j)' (\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j) / n \hat{\lambda}_j \quad (8)$$

and the means and variances of the predictions of \mathbf{y} for each population are:

$$\hat{\mathbf{w}}_j = \sum_{i=1}^n \hat{\tau}_{ij} \mathbf{y}_i / n \hat{\lambda}_j \quad (9)$$

and

$$\hat{\phi}_j^2 = \sum_{i=1}^n \hat{\tau}_{ij} (\mathbf{y}_i - \hat{\mathbf{w}}_j)' (\mathbf{y}_i - \hat{\mathbf{w}}_j) / n \hat{\lambda}_j \quad (10)$$

Finally, the posterior probability, $\hat{\tau}_{ij}$, that data point \mathbf{z}_i is an element of population j is estimated by:

$$\frac{\frac{\hat{\lambda}_j}{\hat{\sigma}_j^l \hat{\phi}_j^m} \exp \left\{ -\frac{1}{2\hat{\sigma}_j^2} \|\mathbf{x}_i - \hat{\boldsymbol{\mu}}_j\|^2 - \frac{1}{2\hat{\phi}_j^2} \|\mathbf{y}_i - \hat{\mathbf{w}}_j\|^2 \right\}}{\sum_{t=1}^k \frac{\hat{\lambda}_t}{\hat{\sigma}_t^l \hat{\phi}_t^m} \exp \left\{ -\frac{1}{2\hat{\sigma}_t^2} \|\mathbf{x}_i - \hat{\boldsymbol{\mu}}_t\|^2 - \frac{1}{2\hat{\phi}_t^2} \|\mathbf{y}_i - \hat{\mathbf{w}}_t\|^2 \right\}} \quad (11)$$

where l is the length of \mathbf{x} and m is the length of \mathbf{y} .

Note that if more basis functions are used than are needed, then one or more of the λ_j will approach zero. The excess basis functions are then removed and the equations for \mathbf{w}_j, σ_j and ϕ_j rearranged to avoid division by zero.

The EM algorithm, which we use to estimate the model parameters, is an iterative approach to maximum likelihood estimation [2] Each iteration of an EM algorithm is composed of two step, which for this problem take the following form: an Expectation (E) step in which, given assumed centers and widths for the basis functions, one determines the probability that each point in the training set comes from each Gaussian (Eqn. 12), and a Maximization (M) step in which the centers and widths of the Gaussians and the means and standard deviations of each of their outputs are estimated, assuming that the assignment of data points to experts is known (Eqns. 7-11). In statistical terms, the M step involves maximization of a likelihood function that is redefined in each iteration of the E step. See Ungar, et al. 1994 for details.

Once the model has been learned, if one knows both \mathbf{x} and \mathbf{y} then equation ?? gives an estimate of the probability that a point \mathbf{z}_i is in the population j . In the more usual case where \mathbf{x} is known and one wishes to estimate \mathbf{y} , one first estimates the probability that \mathbf{x}_i is in population j using Eqn. 3 and then uses the above probability and the estimate of \mathbf{w}_j from Eqn. 12 in Eqn. 2.

2.1. Estimating prediction error

The EMRBF formalism suggests several ways of estimating both the pointwise uncertainties of the estimates of \mathbf{y} and the novelty of new \mathbf{x} 's. The latter is important to know in order to warn the user when the network is being asked to extrapolate or interpolate in regions where little data have been seen.

If one assumes (incorrectly) that the population sizes $\lambda_j n$ are known then it is straightforward to make a

pointwise estimate of the confidence bounds on the estimates of \mathbf{y} using the estimated variances. If the λ_j are known, then $\hat{\mathbf{y}}$ is linear in the \mathbf{w}_j , so we can take the weighted sums of the confidence limits on each \mathbf{w}_j .

The overall uncertainty in the estimate of \mathbf{y} is then

$$\hat{\mathbf{y}}_{CL} = \sum_{j=1}^k p_{ij} (\pm t_{\alpha/2} \phi_j / \sqrt{\lambda_j n - 1}) \quad (12)$$

where $t_{\alpha/2}$ is the appropriate t-statistic and ϕ_j is a vector, all of whose elements are ϕ_j .

This estimate is easy to use, but slightly underestimates the size of the true confidence bounds, as it neglects the uncertainty in the population sizes. Empirically, we have found that it accurately estimates measurement noise in the \mathbf{y} s, but fails to account for the error due to model mismatch. Using more complex formulas one can include the effect of the uncertainties in λ .

One can use similar formulas based on relative likelihoods to calculate the novelty of data points in order to warn when extrapolation beyond historic data is occurring.

2.2. Partially inverting the RBFs

The statistical reformulation of radial basis functions also gives an elegant solution to the long-standing problem of partial inversion of networks. In many problems where neural networks are used, one wishes to determine which input value \mathbf{u} should be selected to order obtain a desired output \mathbf{y} . This situation arises in process control applications where the measurement of the system at time $t + 1$ is given by: $\mathbf{y}_{t+1} = f(\mathbf{y}_t, \mathbf{u}_t)$ Since f is nonlinear, uniquely determining \mathbf{u} is, in general, not possible. In conventional neural networks, gradient descent is often used to iteratively solve for \mathbf{u} .

The EMRBF formulation suggests a simpler approach. One can build a Gaussian mixture model as before but where $\mathbf{z} = (\mathbf{y}_t, \mathbf{u}_t, \mathbf{y}_{t+1})$ Although space precludes a full description, it is straightforward to calculate the expected value of \mathbf{u}_t given \mathbf{y}_t and $\mathbf{y}_{setpoint}$ using an analog to Eqn. 2. Note that this gives a closed form solution; no iteration is required. This approach can easily be built into a more sophisticated controller including feedback.

The above "inversion" gives good results only when the control action \mathbf{u}_t being estimated is a unique function of the current state \mathbf{y}_t and the desired state $\mathbf{y}_{setpoint}$. If multiple \mathbf{u} 's can produce the same responses then the expected value of \mathbf{u}_t given \mathbf{y}_t and $\mathbf{y}_{setpoint}$ is a weighted average of the \mathbf{u} 's. This can

be disastrous: if one can either walk to the left of an obstacle or to the right of it, the average of those two actions need not be a good choice. Fortunately, this problem is easily fixed: if non-uniqueness of the action is suspected, an action should be selected by finding all populations into which the \mathbf{x} and \mathbf{y} fall, predicting the \mathbf{u} for each of the populations, and then selecting the “best” \mathbf{u} , e.g. the one which gives the \mathbf{y} closest to the desired \mathbf{y} in the forward model.

For control problems which require multi-step prediction (e.g. control of unstable systems), this inversion should of course not be used. The forward model should be used in a MPC framework.

3. Results

The advantages and disadvantages of EMRBF over conventional radial basis functions are best understood by examining simple examples.

Consider first the task of approximating a simple step function using only three basis functions. If the \mathbf{x} values are distributed roughly uniformly, then the k-means clustering step of RBFs will create three evenly spaced clusters. The cluster with the smallest \mathbf{x} center will accurately approximate the bottom part of the step and the cluster with the largest \mathbf{x} center will accurately approximate the top part of the step, but the middle cluster will attempt to average \mathbf{y} s from both the top and bottom of the step, and so will end up producing a suboptimal approximation.

In contrast, the EMRBF includes the \mathbf{y} values when clustering, and so produces clusters that are centered on the top and bottom halves of the step. The “extra” cluster ends up converging to one of the two useful clusters and a sharper transition at the step results. In typical results, EMRBF has a Root Mean Squared Error (RMSE) of 0.18, while the RBF has error of 0.22. EMRBF still has some error since it does do some smoothing of the region where the basis functions overlap. This could be reduced by using more basis functions.

Approximating a sine wave shows the disadvantage of the EMRBF. For an approximation to $y = \sin(2\pi x) + \text{noise}$ over one cycle, EMRBF has an error (RMSE) of 0.12 while the RBF has an error of 0.05. (The error is calculated relative to the noise-free sine function to avoid overfitting; the noise level was such that the error fitting the training data to the true sine was 0.09) The above error results largely from the EMRBF model’s assumption that \mathbf{x} is constant on each interval. Using a full covariance matrix in the EMRBF, which allows \mathbf{x} and \mathbf{y} to be linearly correlated

within each cluster, gives an error of 0.06.

Tests on more complex functions such as simulations of chemical reactors show that for smoothly varying plants with low dimensional inputs, RBFs outperform EMRBF. EMRBF offers an advantage (1) when the plant shows discontinuities, as arise from hitting constraints or from control laws such as gain scheduling, (2) when a formal measure of sensor noise or of prediction uncertainty is beneficial (e.g. for controller design) and (3) for multivariate systems where one wishes to calculate the full covariance matrix.

4. Discussion

Radial basis function (RBF) neural networks provide an attractive method for high dimensional nonparametric estimation for use in nonlinear control. They are faster to train than conventional feed forward networks with sigmoidal activation networks (“back-propagation nets”). The RBF model structure is better suited for adaptive control, since if the basis functions are fixed, the model is linear in the coefficients. The RBF model also lends itself to designing fuzzy controllers, since it can be viewed as being a “fuzzy” (linear) combination of simple constant or linear models on different operating regions.

We have shown how radial basis functions can be interpreted as mixtures of Gaussians, and how their parameters can be estimated using the EM algorithm for likelihood maximization. We call this modeling method EMRBF. The EM algorithm is a significant improvement over previous methods of fitting the RBF parameters for certain classes of problems - particularly those with discontinuities, taking advantage of both \mathbf{x} and \mathbf{y} values in the clustering, and giving rapid convergence to a guaranteed local optimum in the centers and widths of the radial basis functions, as well as their coefficients. The mixture model interpretation of EMRBF shows the assumptions implicit in radial basis functions and provides a less ad hoc way of determining the widths of the basis functions than standard methods. By taking more general forms of the covariance between the measured variables, EMRBF is easily extended to include elliptical basis functions and a yet more general class of basis functions in which all input and output variables are correlated with each other.

The statistical formulation also allows us to provide confidence limits on predictions made using the networks and to detect when extrapolation is occurring beyond the region where data were available for training the network. The confidence limits are important for robust controller design, while the extrapolation

warnings are useful when designing safety systems to assure safe performance outside the modeled region.

References

- [1] A.R. Barron. Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, 39(3):930–945, 1993.
- [2] A.P. Dempster, N.M. Laird, and D.B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 93:1–88, 1977.
- [3] F. Girosi and T. Poggio. Networks and the best approximation property, *Biological Cybernetics* 63, 169-176, 1990.
- [4] K.J. Hunt, D. Sbarbaro, R. Zbikowski, and P.J. Gawthrop. Neural networks for control-systems—A survey. *Automatica*, 28(6):1083–1112, 1992.
- [5] M.A. Kramer, M.L. Thompson, and P.M. Bhagat. Embedding theoretical models in neural networks. *Proceedings of the 1992 ACC*. 475–479, 1992.
- [6] J.A. Leonard, M.A. Kramer, and L.H. Ungar. Using radial basis functions to approximate a function and its error bounds. *IEEE Transactions on Neural Networks*. 3:624–627, 1992.
- [7] MacLachlan and Basford. *Mixture Models*. Chapman and Hall, 1988.
- [8] J. Moody and C.J. Darken. Fast learning in networks of locally tuned processing units. *Neural Computation*, 1:281–294, 1989.
- [9] K. Parthasarathy and K. Narendra. *Stable adaptive control of a class of discrete-time nonlinear systems using radial basis neural networks*. Report No. 9103, Electrical Engineering, Yale University, 1991.
- [10] T. Poggio and F. Girosi. Regularization algorithms for learning that are equivalent to multilayer networks. *Science*. 247:978–982, 1990.
- [11] D.C. Psychogios and L.H. Ungar. Direct and indirect model based control using artificial neural networks. *Industrial Engineering Chemical Research*. 30:2564–2573, 1991.
- [12] R.M. Sanner and J.-J.E. Slotine. Gaussian networks for direct adaptive control. *IEEE Trans. Neural Networks*. 3:837-863, 1992.
- [13] L.H. Ungar, T. Johnson and R.D. DeVeaux Radial Basis Functions for Process Control, *CIMPRO Proceedings*. 1994