USC–SIPI REPORT #229

Fuzzy Basis Functions: Comparisons with Other Basis Functions

by

Hyun M. Kim and Jerry M. Mendel

January 1993

Signal and Image Processing Institute
UNIVERSITY OF SOUTHERN CALIFORNIA
Department of Electrical Engineering-Systems
3740 McClintock Avenue, Room 400
Los Angeles, CA 90089-2564 U.S.A.

Fuzzy Basis Functions: Comparisons with Other Basis Functions

Hyun M. Kim and Jerry M. Mendel
Signal and Image Processing Institute
Department of Electrical Engineering - Systems
University of Southern California
Los Angeles, CA 90089-2564

Abstract

Fuzzy basis functions (FBF's) which have the capability of combining both numerical data and linguistic information, are compared with other basis functions. Because a FBF network is different from other networks in that it is the only one that can combine numerical and linguistic information, comparisons are made when only numerical data is available. In particular, a FBF network is compared with a radial basis function (RBF) network from the viewpoint of function approximation. Their architectural interrelationships are discussed. Additionally, a RBF network, which is implemented using a regularization technique, is compared with a FBF network from the viewpoint of overcoming ill-posed problems. A FBF network is also compared with Specht's Probabilistic Neural Network and his General Regression Neural Network (GRNN) from an architectural point of view. This is motivated by the similarities of the FBF and GRNN formulas. Then, a FBF network is compared with a Gaussian sum approximation in which Gaussian functions play a central role. Finally, we summarize the architectural relationships between all the networks discussed in this paper, and compare the different approximations from the point of view of the assumptions made about the available data.

I. Introduction

Recently, Wang and Mendel [30] introduced fuzzy basis functions (FBF's) which have the capability of combining both numerical data and linguistic information. These basis functions are quite general. Their exact mathematical structure depends on three choices that one must make for any fuzzy logic system, namely, type of: membership function, inference mechanism, and defuzzification

strategy. Wang and Mendel frequently choose: Gaussian membership functions, product inference, and centroid defuzzification, in which case their FBF network can be summarized by the following mathematical equation:

$$f(\underline{x}) = \alpha \frac{\sum_{l=1}^{M} \bar{z}^l [\prod_{i=1}^{n} a_i^l exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]}{\sum_{l=1}^{M} [\prod_{i=1}^{n} a_i^l exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]} + \beta \frac{\sum_{l=M+1}^{N} \bar{z}^l [\prod_{i=1}^{n} a_i^l exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]}{\sum_{l=M+1}^{N} [\prod_{i=1}^{n} a_i^l exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)]}, \quad (1)$$

where $\underline{x} = (x_1, \ldots, x_n)^T$; $a_i^l, \bar{x}_i^l$, and σ_i^l are real-valued parameters; $\bar{z}^l, l = 1, \ldots, M$ are coefficients; $\bar{z}^l, l = M+1, \ldots, N$ are the points in the output space R determined by the fuzzy rule base [30]; and, α, β are constants which determine the ratio between numerical data and linguistic information, with $\alpha, \beta \geq 0, \alpha + \beta = 1$. We can represent (1) as a sum of two functions,

$$f(\underline{x}) = \alpha f_N(\underline{x}) + \beta f_L(\underline{x}), \quad (2)$$

where $f_N(\underline{x})$ consists of basis functions which deal with numerical data and $f_L(\underline{x})$ consists of basis functions which deal with linguistic information. Note that in (1), the $\bar{z}^l$ coefficients of $f_L(\underline{x})$ can be obtained from the given fuzzy rule base directly, whereas the $\bar{z}^l$ coefficients of $f_N(\underline{x})$ can be obtained using learning rules such as least squares, least mean-squares, or back-propagation [32].

To people familiar with aspects of approximation theory for deterministic data, the formulas for either $f_N(\underline{x})$ or $f_L(\underline{x})$ look familiar. The radial symmetry of the Gaussian membership functions causes us to wonder whether (1) is just a Gaussian radial basis function expansion (others have suggested to us privately that they are the same). We show below that it is not; instead, it is a nonlinear function of Gaussian radial basis functions. Interestingly enough, Gaussian radial basis functions are themselves special cases of Generalized radial basis functions [8] and hyper basis functions [8]. We also explore the relations between these more general radial basis functions and FBF's below.

There are other literatures in which Gaussian functions play a central role in approximations; hence, we are motivated to explore the relationship between (1) and results from these approximation problems. We were intrigued by the similarity between the formula for Specht's Generalized Regression Neural Network (GRNN) [27] and (1), and wondered again whether (1) was just another GRNN. Below we explain why it is not (see, also, Wang [32]). We also examine the relationship between FBF networks and Specht's Probabilistic Neural Network (PNN) [28], and FBF networks

and Gaussian sum approximations. The latter are used in nonlinear filtering. An important difference between these approximation problems and the ones described in the previous paragraph is that data, including the quantities being estimated by a GRNN, PNN or a Gaussian sum approximation, are assumed from the very beginning to be random. No such modeling assumption is made or needed for FBF networks.

Our comparisons between FBF's and the different types of radial basis functions, or the other functions just described, is only valid for the special case when no linguistic information is used by the FBF network (i.e., $\beta = 0$). We make this important point here, so that it is at once clear to the reader that, in general, a FBF network is indeed different from all of these other networks, because it is the only one that can combine numerical and linguistic information (see, also, Wang[32]). Consequently, the rest of this report treats the special case for a FBF network when only numerical data is available (i.e., $\beta = 0$), in which case it is indeed legitimate to question whether or not the FBF as described by (1) is new.

In Section II, FBF's are compared with a radial basis function (RBF) network which is widely used for interpolation. Because interpolation is a subset of approximation, comparisons are made from the viewpoint of function approximation. Additionally, since the problem of learning a smooth mapping from examples is ill-posed, in the sense that information in the data is not sufficient to uniquely reconstruct the mapping in regions where the data are not available, regularization techniques are compared which change the ill-posed problems into well-posed problems. Finally, the constraints of a regularization technique are interpreted from the viewpoint of linguistic rules.

In Section III, a FBF network is compared with Specht's PNN and his GRNN. Because the PNN and GRNN are based on Parzen's window, which exploits random data, our comparisons are made from an architectural point of view (since the assumptions about the available data are different, it is reasonable to make comparisons based only on network structures).

In Section IV, a FBF network is compared with a Gaussian sum approximation that was developed as early as 1971 to cope with Kalman filtering for non-Gaussian systems.

Finally, in Section V, we present tables which summarize the architectural relationships between the networks discussed in this paper, and the interrelationships of all the networks.

II. Comparison Between FBF's and RBF's

A. FBF Network

The basic configuration of a fuzzy logic system (FLS) is shown in Fig. 1. There are four principal components in a fuzzy logic system: 1) fuzzy rule base which comprises fuzzy rules describing how the fuzzy system performs; 2) fuzzy inference engine which uses the rules in the fuzzy rule base to determine a mapping based on fuzzy logic operations; 3) fuzzifier which maps crisp points in the input space into fuzzy sets in the input space; and, 4) defuzzifier which maps fuzzy sets in the output space into crisp points in the output space. Within each component, there are many different choices that can be made, and many combinations of these choices result in different fuzzy logic systems. It would, therefore, be very cumbersome to compare every case; hence, in this study, we consider only one general case, namely, the fuzzy logic system with: singleton fuzzifier, centroid defuzzifier, product inference, and Gaussian membership function [32], given in (1). We have chosen this case so that the resulting FLS can be compared with seemingly similar systems that also use Gaussian or radially-symmetric functions. Our FLS can be represented as the FBF expansion

$$f(\underline{x})|_{\beta=0} = f_N(\underline{x}) = \sum_{j=1}^{M} p_j(\underline{x})\theta_j \tag{3}$$

where $p_j(\underline{x})$ is a FBF defined as

$$p_j(\underline{x}) = \frac{\prod_{i=1}^{n} \mu_{F_i^j}(x_i)}{\sum_{j=1}^{M} \prod_{i=1}^{n} \mu_{F_i^j}(x_i)} \tag{4}$$

in which $\mu_{F_i^j}(x_i) = a_i^j exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^j}{\sigma_i^j})^2)$ are Gaussian membership functions, and $\theta_j = \bar{z}^j \in R$ are constants. If we fix all the parameters in $p_j(\underline{x})$ at the very beginning of the FBF expansion design procedure, so that the only free design parameters are θ_j, then $f(\underline{x})$ in (3) is linear in the design parameters. The FBF network can then be regarded as a special case of the linear regression model,

$$d(t) = \sum_{j=1}^{M} p_j(t)\theta_j + \epsilon(t) \tag{5}$$

where $d(t)$ is the desired output, θ_j are the unknown parameters, $p_j(t)$ are known as "regressors," which are some fixed functions of $\underline{x}(t)$, and, $\epsilon(t)$ is an error signal which is assumed to be orthogonal to the regressors. By providing input and desired output pairs, the values of the θ_j's can be

determined using, for example, least squares. Note, also, that in general the parameters of the FBF's need not be fixed ahead of time. They can be optimized along with the θ_j's using a back-propagation procedure [31].

B. RBF Network

The RBF network is one of the possible solutions to the real multivariate interpolation problem, that can be stated as follows: given N different points $\{\underline{x}_i \in R^n, i = 1, \ldots, N\}$ and N real numbers $\{y_i \in R | i = 1, \ldots, N\}$, find a function F from R^n to R satisfying the interpolation conditions:

$$F(\underline{x}_i) = y_i, \qquad i = 1, \ldots, N. \tag{6}$$

The RBF approach consists of choosing F from a linear space of dimension N that depends on the data points $\{\underline{x}_i\}$. The basis of this space is chosen to be the set of functions $\{\phi(\| \underline{x} - \underline{x}_i \|), i = 1, \ldots, N\}$ where $\| \cdot \|$ denotes the Euclidean norm. The radially symmetric function $\phi(\cdot)$ which maps from $\mathbf{R}^+$ to $\mathbf{R}$ is called a radial basis function [23], [24]. Some examples of $\phi(\cdot)$ are:

$$\begin{aligned}
&\phi(r) = r, && r > 0 \quad \text{(linear approximation)},\\
&\phi(r) = r^3, && r > 0 \quad \text{(cubic approximation)},\\
&\phi(r) = exp(-r^2/2\sigma^2), && r > 0 \quad \text{(Gaussian approximation)},\\
&\phi(r) = r^2 log r, && r > 0 \quad \text{(thin plate splines)},\\
&\phi(r) = (r^2 + c^2)^{1/2}, && r > 0 \quad \text{(multiquadrics)},
\end{aligned}$$

where r is a positive constant.

The solution to the interpolation problem has the following form [23]:

$$F(\underline{x}) = \sum_{i=1}^{N} \lambda_i \phi(\| \underline{x} - \underline{x}_i \|) \tag{7}$$

where $\underline{x} \in \mathbf{R}^n$ is the input vector; and the λ_i's $(1 \leq i \leq N)$ are parameters. When the given sample values are presumed to be accurate, and it is required to perform a smooth interpolation between sample points, Eq. (7) can be solved by imposing the interpolation conditions $F(\underline{x}_j) = y_j$, $j = 1, \ldots, N$. The solution is

$$\underline{\Lambda} = \Phi^{-1} \underline{y} \tag{8}$$

where

$$\underline{\Lambda}_{N \times 1} = [\lambda_1, \ldots, \lambda_N]^T \tag{9}$$

$$(\Phi)_{ij} = \phi(\| \underline{x}_i - \underline{x}_j \|) \tag{10}$$

and

$$\underline{y}_{N\times 1} = [y_1, \ldots, y_N]^T \tag{11}$$

Some analysis regarding the singularity of Eq. (8) is made by Micchelli [16].

If the data are subject to measurement errors or stochastic variations, a strict interpolation is meaningless. Consequently, the interpolation property of the RBF network is not sufficient to guarantee good results. One of the solutions to this problem is the regularization technique which exploits smoothness constraints [8], [19] [20]. It consists of replacing the matrix Φ by $\Phi + \alpha I$, where I is the identity matrix, and α is a small parameter whose magnitude is proportional to the amount of noise in the data points. The coefficients of the RBF network are then given by

$$\underline{\Lambda} = (\Phi + \alpha I)^{-1}\underline{y} \tag{12}$$

Note that the original interpolation is recovered by letting α go to zero. Interpolation is the limit of approximation when there is no noise in the data. It is proved in [19] that, for networks derived from regularization, and in particular for radial basis function networks, a best approximation exists which guarantees that the approximation problem has a unique solution.

C. Comparisons

If we choose the parameters in (1) as $a_i^l = 1$ and $\sigma_i^l = \sigma$ for all $i = 1, 2, ..., n$ and $l = 1, 2, ..., M$, with $\beta = 0$, then

$$f(\underline{x}) = \frac{\sum_{l=1}^{M} \bar{z}^l exp[-\frac{(\underline{x}-\bar{\underline{x}}^l)^T(\underline{x}-\bar{\underline{x}}^l)}{2\sigma^2}]}{\sum_{l=1}^{M} exp[-\frac{(\underline{x}-\bar{\underline{x}}^l)^T(\underline{x}-\bar{\underline{x}}^l)}{2\sigma^2}]} = \frac{\sum_{l=1}^{M} \bar{z}^l exp[-\frac{\|\underline{x}-\bar{\underline{x}}^l\|^2}{2\sigma^2}]}{\sum_{l=1}^{M} exp[-\frac{\|\underline{x}-\bar{\underline{x}}^l\|^2}{2\sigma^2}]}. \tag{13}$$

If we use the Gaussian radial basis function notation, then (13) can be expressed as

$$f(\underline{x}) = \frac{\sum_{l=1}^{M} \bar{z}^l \phi(\| \underline{x} - \bar{\underline{x}}^l \|)}{\sum_{l=1}^{M} \phi(\| \underline{x} - \bar{\underline{x}}^l \|)} \tag{14}$$

Comparing (14) with (7) from an architectural point of view, we see that, whereas the RBF network is a linear combination of radial basis functions, the FBF network is a non-linear combination of radial basis functions; hence, a RBF network can be used as a FBF network with the addition of lateral connections between the RBF's [17]. Due to the centroid defuzzification operation of a FBF network, which leads to the denominator of (14), it is impossible to classify a FBF as a RBF.

After determining the network structure, it is natural to ask how to determine the coefficients of (7) or (14). The computation of the coefficients of the RBF network becomes a very time consuming job as N becomes large. To overcome this, several methods have been proposed. Chen [7] uses an orthogonal least squares(OLS) algorithm to select a subset of significant basis functions from a given set of basis functions. Poggio and Girosi [8] proposed a generalized RBF(GRBF) network which has movable centers that do not necessarily coincide with some of the data points $\underline{x}_i$. They applied the regularization technique to the approximation problem. It consists of looking for the function f that minimizes the functional

$$H[f] = \sum_{i=1}^{N}(y_i - f(\underline{x}_i))^2 + \alpha \parallel Pf \parallel^2 \tag{15}$$

where P is a constraint operator, $\| \cdot \|^2$ is a norm on the function space to which f belongs and α is a regularization parameter. The structure of the operator P embodies the a priori knowledge about the solution. Poggio and Girosi [8] also extended the GRBF to a hyper basis function (HyperBF) network by choosing a different smoothing parameter for each basis function (e.g., a different σ for each Gaussian RBF). The main idea is to consider the mapping to be approximated by the sum of several functions, each one with its own prior, that deals with different constraints to stabilize a system. Consequently a FBF network with different σ's becomes a nonlinear combination of HyperBF's. In other words, a HyperBF network can also be used as a FBF network with the addition of lateral connections between the HyperBF's.

From the point of view of learning as approximation, the problem of learning a smooth mapping from examples is ill-posed in the sense that the information in the data is not sufficient to uniquely reconstruct the mapping in regions where the data are not available. A priori assumptions about the mapping are needed to make the problem well-posed. In particular, the mapping may be smooth, which is one of the most general constraints. We may regard this constraint as a linguistic rule (although it is not an IF-THEN rule). Poggio and Girosi therefore absorb linguistic rules using a constraint operator, whereas a FBF network absorbs linguistic rules directly into its basis functions.

The most important advantage of the FBF network over other networks is that linguistic IF-THEN rules can be translated into FBF's to make the FBF network an universal approximator. To cope with an ill-posed problem, the number of training data must be large enough to excite all the modes of the system. FBF's can incorporate some linguistic IF-THEN rules which play the

role of unobserved modes.

III. Comparison Between FBF Networks and Networks for Which the Data are Assumed to be Random

A. Parzen's Estimate of a Probability Density Function

Central to non-linear estimation and stochastic control problems is the determination of the probability density function of the state conditioned on the available measurement data. If this a posteriori density function is known, then an estimate of the state for any performance criterion can be determined. Parzen [18] showed how one may construct a family of estimates, $f_n(x)$, of a probability density function (PDF) $f(x)$, as

$$f_n(x) = \frac{1}{n\sigma} \sum_{i=1}^{n} k(\frac{x - X_i}{\sigma}), \tag{16}$$

which is consistent at all points x at which the PDF is continuous. Let $X_1, \ldots, X_n$ be independent random variables identically distributed as a random variable X whose distribution function $F(x) = P[X \leq x]$ is absolutely continuous. Parzen's conditions on the weighting function $k(y)$ are

$$sup_{-\infty<y<\infty} |k(y)| < \infty, \tag{17}$$

where *sup* indicates the supremum,

$$\int_{-\infty}^{\infty} |k(y)| \, dy < \infty, \tag{18}$$

$$\lim_{y \to \infty} |yk(y)| = 0, \tag{19}$$

and

$$\int_{-\infty}^{\infty} k(y) \, dy = 1. \tag{20}$$

In Eq. (16), $\sigma = \sigma(n)$ is chosen as a function of n such that

$$\lim_{n \to \infty} \sigma(n) = 0, \tag{21}$$

and

$$\lim_{n \to \infty} n\sigma(n) = \infty. \tag{22}$$

Parzen proved that the estimate $f_n(x)$ is consistent in the mean-squared sense in that

$$E\{|f_n(x) - f(x)|^2\} \rightarrow 0 \qquad as \quad n \rightarrow \infty. \tag{23}$$

Cacoullos [6] has extended Parzen's results to cover the multivariate case. In the particular case of the Gaussian kernel, the multivariate estimates can be expressed as

$$f_m(\underline{x}) = \frac{1}{(2\pi)^{p/2}\sigma^p}\frac{1}{m}\sum_{i=1}^{m} exp[-\frac{(\underline{x} - \underline{X_i})^T(\underline{x} - \underline{X_i})}{2\sigma^2}] \tag{24}$$

where i = pattern number, m = total number of training patterns, $\underline{X_i}$ = ith training pattern, σ = smoothing parameter, and p = dimensionality of measurement space, i.e., dim($\underline{x}$). Observe that $f_m(\underline{x})$ looks like (7) in which all the λ_i's are the same; hence, in retrospect, Parzen's PDF, $f_m(\underline{x})$ can now be called a RBF network. Except for a new name, Parzen's important result remains unchanged. *Note, though, that in Parzen's work signal x is random, whereas RBF's do not assume the data is random.*

Although Parzen proved the existence of a consistent estimate in mean-square, he did not indicate how to choose the weighting function on the basis of a finite set of data. Several methods have been proposed for the practical use of Parzen's method. Breiman *et al.* [4] suggested that even better density estimates could be obtained using Parzen windows and finite data sets if a different σ is used for each exemplar (data point). Their suggestion stems from the observations that: Parzen's method can not respond appropriately to variations in the PDF (i.e., there should be a distinction between low density regions and high density regions); and, none of the asymptotic results give any helpful leads on how the shape factor σ should be selected to give the best estimate of the unknown density. In other words, the rate of convergence depends critically on the density and its derivatives. To make the sharpness of the kernel data-responsive, they proposed the class of estimates

$$f_n(\underline{x}) = \frac{1}{n}\sum_{i=1}^{n}(\alpha_k d_{i,k})^{-m} K(\frac{\underline{x} - \underline{x_i}}{\alpha_k d_{i,k}}) \tag{25}$$

where $d_{i,k}$ is the distance from the point $\underline{x_i}$ to its kth nearest neighbor, α_k is a constant multiplicative factor, m is the dimension of $\underline{x}$ and K is a selected kernel. Observe that in low density regions, $d_{i,k}$ will be large and the kernel will be spread out. This method can be regarded as an extended version of the kth nearest neighbor estimator [12] which is adaptive to local sample density, but is discontinuous. The variable kernel approach offers a combination of the desirable smoothness properties of the Parzen-type estimators with the data-adaptive character of the kth nearest neighbor

approach. Breiman *et al.*, observed that the best value of σ for the Parzen estimator depends on which measure of error is used, and hence would be much more difficult to use in practice than the variable kernel method when the PDF is unknown. They concluded, through some simulations, that the variable kernel estimate was superior to the Parzen estimate. The main disadvantage of the variable kernel estimate is its lack of systematic learning rules. It is based on a rule which tries to find the best value of k by varying it from an initial guess.

Equation (1) can also be regarded as a variable kernel estimate if we do not fix the value of σ's ahead of time; but, *as in the case of RBF's, the FBF's in (1) do not assume the data is random.*

B. Probabilistic Neural Networks

Specht's PNN [28] is based on a non-parametric estimation of a probability density function, so that a Bayes decision rule can be used for pattern classification. Consider the two-category situation in which the state of nature S is known to be either S_A or S_B. Let the measurements be represented by the p-dimensional vector $\underline{x} = [x_1, \ldots, x_p]^t$; then, the Bayes decision rule becomes

$$d(\underline{x}) = S_A \quad if \quad h_A l_A f_A(\underline{x}) > h_B l_B f_B(\underline{x}) \tag{26}$$

$$d(\underline{x}) = S_B \quad if \quad h_A l_A f_A(\underline{x}) < h_B l_B f_B(\underline{x}) \tag{27}$$

where $f_A(\underline{x})$ and $f_B(\underline{x})$ are the PDF's for categories A and B, respectively; l_A is the loss function associated with the decision $d(\underline{x}) = S_B$ when $S = S_A$; l_B is the loss function associated with the decision $d(\underline{x}) = S_A$ when $S = S_B$; h_A is the a priori probability of occurrence of patterns from category A; and $h_B = 1 - h_A$ is the a priori probability that $S = S_B$.

The boundary between the region in which the Baye's decision $d(\underline{x}) = S_A$ and the region in which $d(\underline{x}) = S_B$ is given by the equation

$$f_A(\underline{x}) = K f_B(\underline{x}) \tag{28}$$

where

$$K = h_B l_B / h_A l_A \tag{29}$$

The key to using Eq. (28) is the ability to estimate PDF's based on training patterns. Specht's probabilistic neural network uses Parzen's method to estimate the PDF, consequently, if we use Gaussian functions for the weighting function of (16), it becomes a RBF network. However, when

this approach was first proposed and used for pattern recognition, there were two limitations inherent in the use of Parzen's method: (1) the entire training set must be stored and used during testing; and, (2) the amount of computation necessary to classify an unknown point is proportional to the size of the training set. Both considerations severely limited the direct use of Parzen's method in real-time. To overcome these limitations, Specht proposed polynomial discriminant functions [29], which approximated Parzen's estimates, using Taylor series, to reduce the number of calculations. With the advent of VLSI technology, Specht implemented the PNN using Parzen's method without simplification [28].

A technique similar to Specht's polynomial discriminant functions was explored by other researchers [1], [3], [14], who referred to their work using the term "potential functions". This term first appeared in the pattern recognition literature when the Soviets [1] introduced a simple algorithm of potentials. Their method was originally suggested by the idea that, if data samples are thought of as points in a multidimensional space, and if electrical charges are placed at these points, the electrostatic potential would serve as a useful discriminant function.

C. General Regression Neural Networks

Specht [27] also extended his PNN to a GRNN. A more general approach to forming an associative memory is to avoid distinguishing between inputs and outputs. By concatenating the input vector and the output vector into one longer measurement vector, the joint PDF can be obtained. Let $f(\underline{x}, z)$ be the joint probability density function of a random vector, $\underline{x} \in R^n$, and a random variable, $z \in R$. The conditional mean of z given $\underline{x}$ (also called a mean-squared estimator) is given by

$$E[z|\underline{x}] = \int_{-\infty}^{\infty} z f(z|\underline{x}) dz = \frac{\int_{-\infty}^{\infty} z f(\underline{x}, z) dz}{\int_{-\infty}^{\infty} f(\underline{x}, z) dz}. \tag{30}$$

Let $(\underline{\bar{x}}^l, \bar{z}^l)$, $l = 1, 2, ..., N$, be sample values of the random variables $\underline{x}$ and z; then, a consistent estimator of $f(\underline{x}, z)$, based upon Parzen's method, is given from Eq. (24) as

$$\hat{f}(\underline{x}, z) = \frac{1}{(2\pi)^{(n+1)/2}\sigma^{(n+1)}} \frac{1}{N} \sum_{l=1}^{N} exp[-\frac{(\underline{x} - \underline{\bar{x}}^l)^T(\underline{x} - \underline{\bar{x}}^l)}{2\sigma^2}] exp[-\frac{(z - \bar{z}^l)^2}{2\sigma^2}]. \tag{31}$$

Substituting (31) into (30) and performing the integration yields the following:

$$z(\underline{x}) = \hat{E}(z|\underline{x}) = \frac{\sum_{l=1}^{N} \bar{z}^l exp[-\frac{(\underline{x}-\underline{\bar{x}}^l)^T(\underline{x}-\underline{\bar{x}}^l)}{2\sigma^2}]}{\sum_{l=1}^{N} exp[-\frac{(\underline{x}-\underline{\bar{x}}^l)^T(\underline{x}-\underline{\bar{x}}^l)}{2\sigma^2}]}, \tag{32}$$

which is the probabilistic general regression used in Specht's GRNN.

D. Comparisons

Comparing (32) with (1), we see that they are almost the same. If we choose the parameters in (1), as: $M = N, a_i^l = 1, \sigma_i^l = \sigma$ for all $i = 1, 2, ..., n$ and $l = 1, 2, ..., M$, $\bar{x}_i^l$ = the ith element of the sample vector $\underline{\bar{x}}^l$, and, $\bar{z}^l$ = the sample $\bar{z}^l$, and $\beta = 0$, then the equation for the fuzzy system (1) becomes structurally the same as the equation for the probabilistic general regression (32). In this case, it seems that the centroid defuzzifier makes the FBF network play the role of a mean-squared estimator. Note, also, that the product inference rule makes the multivariate kernel a product of univariate kernels. Poggio and Girosi [19], [20] also showed the relation between regularization and Bayes estimation, and demonstrated that Eq. (15) coincides with maximum a posteriori (MAP) estimation provided that the noise is additive and Gaussian, and the prior is Gaussian. Note that the mean-squared estimator gives the same result as the MAP estimator when measurements and unknown parameters are jointly Gaussian [15].

Although *there is no statistical consideration when constructing a fuzzy system,* if we apply Parzen's conditions [(17)-(22)] to fuzzy membership function (MF) $\mu_F : U \rightarrow [0, 1]$, they can be easily satisfied by relaxing the constraint that the range of the membership function satisfies Eq. (20). Equations (21) and (22) guide the choice of the smoothing parameter, i.e., σ in the special case of (1).

We can also apply Eq. (30) to a triangular weighting function, $k(y)$, one that satisfies Parzen's conditions. In this case:

$$\hat{f}(\underline{x}, z) = \frac{1}{(n+1)} \frac{1}{N} \sum_{l=1}^{N} [\prod_{i=1}^{n} \wedge(\frac{x_i - \bar{x}_i^l}{\sigma})] \wedge (\frac{z - \bar{z}^l}{\sigma}) \tag{33}$$

and

$$z(\underline{x}) = \hat{E}(z|\underline{x}) = \frac{\sum_{l=1}^{N} \bar{z}^l \prod_{i=1}^{n} \wedge(\frac{x_i - \bar{x}_i^l}{\sigma})}{\sum_{l=1}^{N} \prod_{i=1}^{n} \wedge(\frac{x_i - \bar{x}_i^l}{\sigma})} = \frac{\sum_{l=1}^{N} \bar{z}^l \prod_{i=1}^{n} \wedge(\frac{x_i - \bar{x}_i^l}{2})}{\sum_{l=1}^{N} \prod_{i=1}^{n} \wedge(\frac{x_i - \bar{x}_i^l}{2})} \tag{34}$$

where $\wedge(\frac{x_i - \bar{x}_i^l}{\sigma})$ denotes a triangular function centered at $\bar{x}_i^l$ with base σ and height $\frac{2}{\sigma}$. In Eq. (1), if we use the triangular membership function with same σ's, we again obtain (34); therefore, the resulting fuzzy system again turns out to be a GRNN.

Although a FBF network and GRNN are similar in special situations, they are quite different from many fundamental points of view. For example, FBF networks are constructed from a combination of sample data pairs $(\underline{\bar{x}}^l, \bar{z}^l)$, and fuzzy IF-THEN rules, whereas the GRNN is constructed

only from the sample data pairs $(\underline{x}^l, z^l)$. Additionally, the data is assumed to be random for a GRNN, whereas no such assumption is made or needed for a FBF network. Whereas FBF networks provide a very good framework to combine linguistic information and measured numerical information , the GRNN can only make use of the numerical information.

In this section, we showed that a PNN is a special case of RBF networks. We, also showed that the structure of a GRNN is a special case of FBF networks, i.e., given the same set of information (sample pairs), we can construct a fuzzy system (using fuzzy logic principles) which has exactly the same structure as the GRNN, by modifying the range of the MF. This can be justified by Parzen's conditions. Also, we can speculate that the sharper the shape of the MF, the stronger is our belief in a fuzzy set. Wang [32] mentioned this point and claimed that his modified centroid defuzzifier, which exploits this speculation by modifying the range of the MF, would result in better performance than the centroid defuzzifier.

The principle advantages of GRNN's are fast learning and convergence to the optimal regression surface as the number of samples become very large. The disadvantage of GRNN's is the amount of computation required of the trained system to estimate a new output vector. Burrascano [5] has suggested using learning vector quantization to find representative samples, to reduce the size of the training set for a GRNN. Schioler and Hartmann [25] proposed an algorithm to alleviate the computational burden based on the ideas for automatic recruitment of new centers. The OLS learning algorithm [7] also overcomes this problem to a large degree by selecting a subset of significant regressors using projections.

IV. Comparison Between FBF's and Gaussian Sum Approximations

A. Gaussian Sum Approximations

Gaussian sum approximations have been proposed as a means to accomplish practical nonlinear Bayesian filtering [2], [26]. They are motivated by the fact that the Kalman filter, which is valid only for linear Gaussian systems, continues to be widely (and heuristically) used for non-linear or non-Gaussian systems.

Consider a probability density function $f(x)$. The problem of approximating $f(x)$ can be

conveniently considered within the context of delta families of positive type [10]. Using the delta families, the following result can be used for the approximation of a density function $f(x)$.

Theorem [10]: Let δ_σ belong to a delta family whose limit function behaves like a delta function; then the sequence $f_\sigma(x)$ which is formed by the convolution of δ_σ and f, as

$$f_\sigma(x) = \int_{-\infty}^{\infty} \delta_\sigma(x-u) f(u) du, \tag{35}$$

converges uniformly to $f(x)$ on every interior subinterval of $(-\infty, \infty)$.

When f has a finite number of discontinuities, the theorem is still valid except at the points of discontinuity. If δ_σ is required to satisfy the condition that

$$\int_{-\infty}^{\infty} \delta_\sigma(x) dx = 1, \tag{36}$$

it follows from Eq. (35) that f_σ is a probability density function for all σ; therefore, in Kalman filtering, the following delta family is a natural choice for density approximations:

$$\delta_\sigma(x) \triangleq N_\sigma(x) = (2\pi\sigma^2)^{-\frac{1}{2}} exp[-\frac{x^2}{2\sigma^2}] \tag{37}$$

It is shown in [21] that the Gaussian density tends to the delta function as the variance tends to zero. Using Eq. (37), the density approximation f_σ is written as

$$f_\sigma(x) = \int_{-\infty}^{\infty} N_\sigma(x-u) f(u) du \tag{38}$$

It is this form that provides the basis for the Gaussian sum approximations. It is clear that $N_\sigma(x-u)f(u)$ is integrable on $(-\infty, \infty)$ and is at least piecewise continuous; thus, (38) can itself be approximated on any finite interval by a Riemann sum. Consequently, an approximation of f_σ over some bounded interval (a, b) can be written as

$$f_{m,\sigma}(x) = \sum_{i=1}^{m} \alpha_i N_\sigma(x - x_i). \tag{39}$$

For practical purposes, it is desirable that f be approximated to within an acceptable accuracy by a relatively small number of terms of the series. For the subsequent discussion, it is convenient to write the Gaussian sum approximation as

$$f_n(x) = \sum_{i=1}^{n} \alpha_i N_{\sigma_i}(x - \mu_i) \tag{40}$$

where $\sum_{i=1}^{n} \alpha_i = 1$, and $\alpha_i \geq 0$ for all i, and $n \leq m$ in (39). Unlike Eq. (39), it has been assumed in (40) that the variance σ_i^2 can vary from one term to another, in order to obtain greater flexibility for approximations using a finite number of terms. Certainly, as the number of terms increase, it is necessary to require that σ_i tend to become equal and vanish.

The problem of choosing the parameters α_i, μ_i, and σ_i to obtain the best approximation f_n to some density function has been considered [13], [17]. In many problems, it may be desirable to cause the approximation to match some of the moments. For example, if the mean associated with f is μ, then the constraint that f_n have mean value μ would be

$$\mu = \int_{-\infty}^{\infty} x f_n(x) dx = \int_{-\infty}^{\infty} x[\sum_{i=1}^{n} \alpha_i N_{\sigma_i}(x - \mu_i)] dx \tag{41}$$

or,

$$\mu = \sum_{i=1}^{n} \alpha_i \mu_i. \tag{42}$$

Several methods [9], [17] have been developed for choosing the parameters α_i, μ_i, and σ_i. Let P denote the Gaussian representation parameters given by

$$P = [(\alpha_i \;\; \mu_i \;\; \sigma_i), i = 1, 2, \ldots, n]. \tag{43}$$

The optimal Gaussian sum approximation [9] is obtained by minimizing the sum of the squared errors between samples of the original signal and the approximation signal with respect to the parameters P; i.e.,

$$\hat{P} = \arg \min_{P} E, \tag{44}$$

where

$$E = \sum_{i=1}^{N} [f(x_i) - f_n(x_i)]^2. \tag{45}$$

The solution of (45) may be derived by solving the system of nonlinear equations obtained by setting $\partial E / \partial P_l = 0$, for $l = 1, 2, \ldots, 3n$ (the $3n$ unknown parameters in P). The steepest-descent method is a commonly used approach to the solution of nonlinear minimization problems. While it guarantees local convergence, it poses some restrictions in learning rate, and, converges slowly. Some modified algorithms are proposed in [9] to obtain an iterative optimization procedure which results in fast convergence.

Generally, the performance of the gradient descent method is strongly dependent on the choice of the initial parameters. Additionally, the number of Gaussian basis functions also affects the approximation error. In [9], a scale-space image of the signal is used to estimate these parameters.

B. Comparisons

If we extend the 1-D Gaussian sum approximation in (40) to the representation of m-dimensional signals, then

$$\hat{f}(\underline{x}) = \sum_{l=1}^{n} \bar{z}^l [\prod_{i=1}^{m} exp(-\frac{1}{2}(\frac{x_i - \bar{x}_i^l}{\sigma_i^l})^2)] \tag{46}$$

Comparing (46) with (7), we see that they are almost the same. If we let $\sigma_i^l = \sigma$ for all $i = 1, 2, \ldots, n$ they are exactly the same. Hence, the Gaussian sum approximation can be regarded as an extended version of a RBF network where the radial basis function is Gaussian. It can also be viewed as a class of variable kernel estimates.

V. Conclusions

Table 1: Architectural comparisons of FBF network with others, when no linguistic information is used by the FBF network.

amplitudes of MF's (a_i's)	center locations of MF's ($\underline{x}_i$'s)	variances of MF's (σ_i's)	corresponding network
fixed	fixed	fixed	—
fixed	fixed	variable	—
fixed	variable	fixed	GRNN
fixed	variable	variable	—
variable	fixed	fixed	—
variable	fixed	variable	—
variable	variable	fixed	—
variable	variable	variable	—

In this study, we showed that a FBF network is a non-linear combination of RBF's or HyperBF's depending on the value of σ. We interpreted the constraints of regularization techniques as linguistic rules to compare how they are utilized in the function approximation and showed that Poggio and Girosi absorb linguistic rules using a constraint operator, while a FBF network absorbs linguistic rules directly into its basis functions. We also showed that the *structure* of a GRNN is a special case of a FBF network, although the data is assumed to be random for a GRNN, whereas it is non-random for a FBF network. We also showed that the Gaussian sum approximation can be regarded as an extended version of a Gaussian RBF network.

In Table 1, we summarize architectural comparisons of a FBF network with other networks, and in Table 2, we summarize architectural comparisons of a RBF network with other networks. In Fig. 2, we summarize the relationships between a RBF network (including the GRBF and the HyperBF networks) and a FBF network which are based on non-random numerical data, as well as the relationship between the Parzen type networks (PNN, GRNN) and the Gaussian sum approximation from the point of view of the assumptions made about the available data.

From the tables, and Fig. 2, we conclude that: (1) FBF's are architecturally different from RBF's and are the only basis functions that can handle linguistic information as well as non-random numerical data; (2) RBF's encompass other techniques such as Gaussian sum approximations; and (3) FBF's are applied to data (and rules) without an underlying assumption of randomness about the data. How to apply FLS's to "random" data is a subject for future study.

Table 2: Architectural comparisons of RBF network with others.

amplitudes of Gaussian functions	$\underline{x}_i$'s	σ_i's	corresponding network
fixed	fixed	fixed	—
fixed	fixed	variable	—
fixed	variable	fixed	GRBF; Parzen's PDF approx.; PNN; Gaussian sum approx.
fixed	variable	variable	Hyper BF; Gaussian sum approx.
variable	fixed	fixed	—
variable	fixed	variable	—
variable	variable	fixed	—
variable	variable	variable	Breiman's extension of Parzen's PDF approx. (σ_i's and a_i's are dependent)

References

[1] M. A. Aizerman, E. M. Braverman, and L. I. Rozonoer, "Theoretical foundations of potential function method in pattern recognition," *Avtomatika i Telemekhanika*, Vol. 25, pp. 971-936, May, 1964.

[2] D. L. Alspach and H. W. Sorenson, "Nonlinear Bayesian estimation using Gaussian sum approximations," *IEEE Trans. Automatic Control*, Vol. AC-17, No. 4, pp. 439-448, Aug. 1972.

[3] E.M. Braverman, "On the method of potential functions," *Avtomatika i Telemekhanika*, Vol. 26, No. 12, pp. 2130-2138, December, 1965.

[4] L. Breiman, W. Meisel, and E. Purcell, "Variable kernel estimates of multivariate densities," *Technometrics*, Vol. 19, No. 2, pp. 135-144, May 1977.

[5] P. Burrascano, "Learning vector quantization for the probabilistic neural network," *IEEE Trans. on Neural Networks*, Vol. 2, pp. 458-461, July 1991.

[6] T. Cacoullos, "Estimation of a multivariate density," *Annals of the Institute of Statistical Mathematics (Tokyo)*, Vol. 18, No. 2, pp. 179-189, 1966.

[7] S. Chen, C. F. N. Cowan and P. M. Grant, "Orthogonal least squares learning algorithm for radial basis function networks," *IEEE Trans. on Neural Networks*, Vol. 2, No. 2, pp.302-309,1991.

[8] F. Girosi and T. Poggio, "Networks and the best approximation property," A.I. Memo No. 1164, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[9] A. Goshtasby and D. Schonfeld, "Signal representation based on a Gaussian decomposition," *Proc. Conf. of the 1991 Conf. on Information Sciences and Systems* (Baltimore,MD), pp. 1-6, 1991.

[10] J. Koreyaar, *Mathematical Methods*, Vol. 1, pp. 330-333, Academic Press, New York, 1968.

[11] R. R. Lemke, "On the application of the potential function method to pattern recognition and system identification," *Ph.D. dissertation*, Purdue Univ., April, 1968.

[12] P. O. Looftgaarden and C. P. Quesenberry, "A nonparametric estimate of a multivariate probability density function," in *Ann. Math. Statist.*, Vol. 28, pp. 1049-1051, 1965.

[13] D. W. Marquardt, "An algorithm for least-squares estimation of nonlinear parameters," *Journal of SIAM*, Vol. 11, No. 2, pp. 431-441, 1963.

[14] W. S. Meisel, "Potential functions in mathematical pattern recognition," *IEEE Trans. on Computers*, Vol. C-18, No. 10, pp. 911-918, 1969.

[15] J. M. Mendel, *Lessons in Digital Estimation Theory*, Prentice-Hall, 1987.

[16] C. A. Micchelli, "Interpolation of scattered data: Distance matrices and conditionally positive definite functions," *Constr. Approx.*, Vol. 2, pp.11-22,1989.

[17] J. Moody and C. J. Darken, "Fast learning in Networks of locally-tuned processing units," *Neural Comp.*, Vol. 1, pp. 281-294, 1989.

[18] E. Parzen, "On estimation of a probability density function and mode," *Ann. Math. Statist.*, Vol. 33, pp. 1065-1076, 1962.

[19] T. Poggio and F. Girosi, "A theory of networks for approximation and learning," A.I. Memo No. 1140, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, 1989.

[20] T. Poggio and F. Girosi, "Networks for approximation and learning," *Proceedings of the IEEE*, Vol. 78, No. 9, pp. 1481-1497, Sept., 1990.

[21] A. D. Poularikas and S. Seely, *Signals and Systems*, PWS-KENT, 2nd-ed., 1991.

[22] M. J. D. Powell, *Approximation Theory and Methods*, Cambridge University Press, Cambridge, 1981.

[23] M. J. D. Powell, "Radial basis functions for multivariable interpolation: A review," in *Algorithms for Approximation*, J. C. Mason and M. G. Cox, Eds., Oxford, pp. 143-167, 1987.

[24] M. J. D. Powell, "Radial basis function approximations to polynomials," in *Proc. 12th Biennial Numerical Analysis Conf.*(Dundee), pp. 223-241, 1987.

[25] H. Schioler and U. Hartmann, "Mapping neural network derived from the Parzen window estimator," *Neural Networks*, Vol. 5, pp. 903-909, 1992.

[26] H. W. Sorenson and D. L. Alspach, "Recursive Bayesian estimation using Gaussian sums," *Automatica*, Vol. 7, No. 4, pp. 465-479, July 1971.

[27] D. F. Specht, "A general regression neural network," *IEEE Trans. on Neural Networks*, Vol. 2, No. 6, pp. 568-576, 1991.

[28] D. F. Specht, "Probabilistic neural networks," *Neural Networks*, Vol. 3, pp. 109-118, 1990.

[29] D. F. Specht, "Generation of polynomial discriminant functions for pattern recognition," *IEEE Trans. Electron. Comput.*, Vol. EC-16, pp. 308-319, 1967.

[30] L. X. Wang and J. M. Mendel, "Fuzzy basis functions, universal approximation, and orthogonal least squares learning," *IEEE Trans. on Neural Networks*, Vol. 3, No. 5, pp. 807-814, Sept., 1992.

[31] L. X. Wang and J. M. Mendel, "Back-propagation fuzzy systems as nonlinear dynamic system identifiers ," *Proc. IEEE International Conf. on Fuzzy Systems*, pp. 1409-1418, San Diego, 1992.

[32] L. X. Wang, "Analysis and design of fuzzy systems," *Ph.D. dissertation*, Univ. of Southern California, April, 1992.

[33] L. X. Wang and J. M. Mendel, "Generating fuzzy rules from numerical data, with applications," to appear in *IEEE Trans. on Systems, Man, and Cybern.*, 1992.

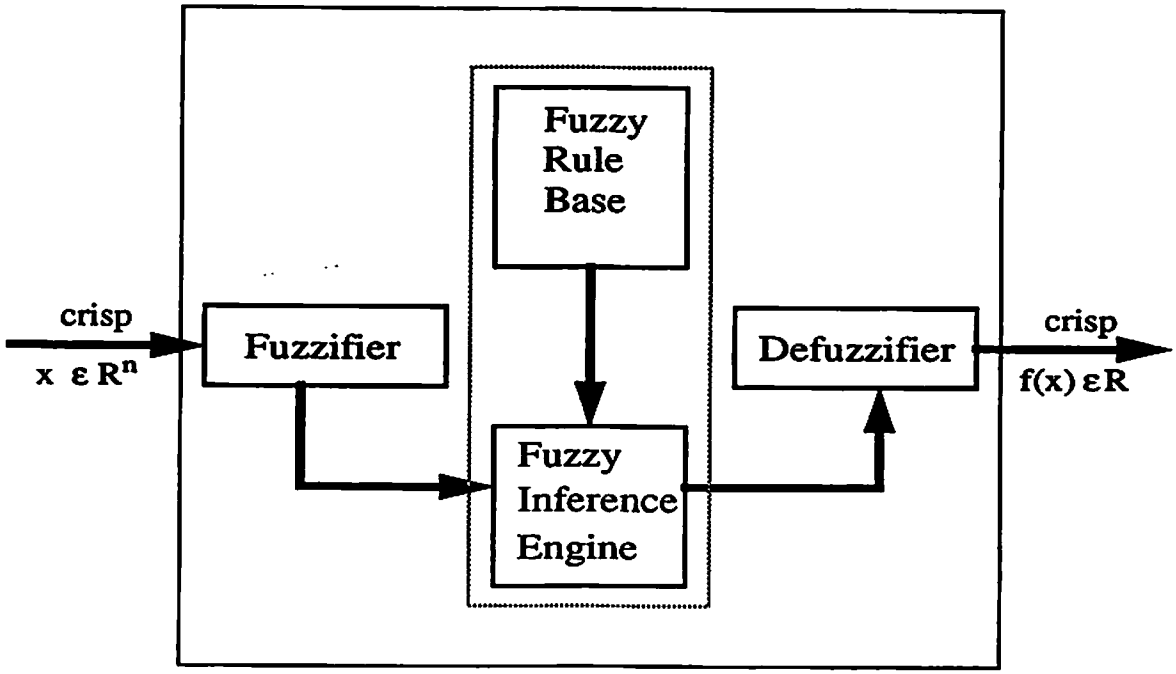

Figure 1: Basic configuration of fuzzy logic system.

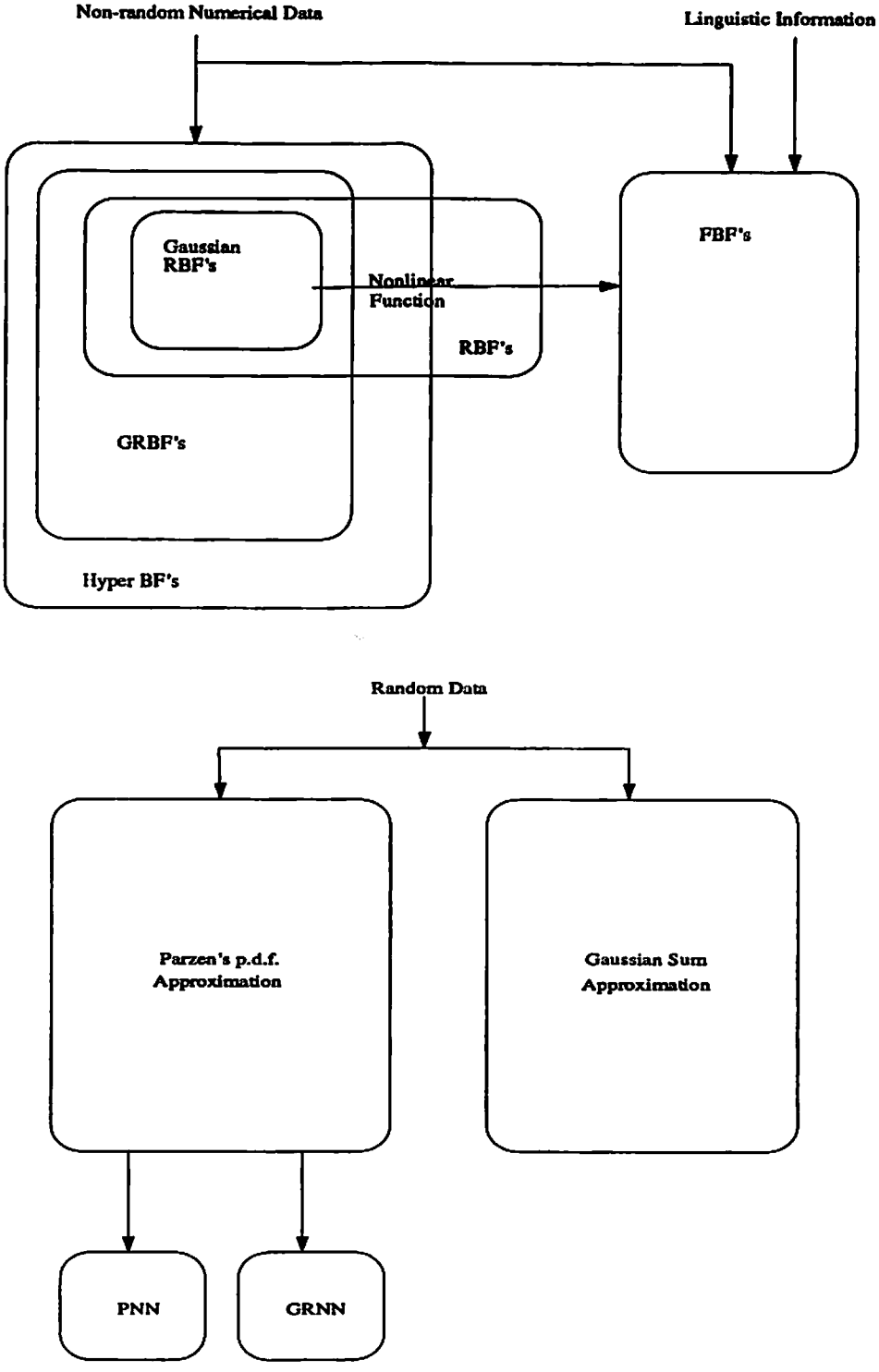

Figure 2: Comparison of different approximations from the point of view of the assumptions made about the available data.