Available online at www.sciencedirect.com
SCIENCE DIRECT®
ELSEVIER
International Journal of Solids and Structures 43 (2006) 3255–3291
INTERNATIONAL JOURNAL OF SOLIDS and STRUCTURES
www.elsevier.com/locate/ijsolstr

Structural reliability analysis for implicit performance function using radial basis function network

Jian Deng *

School of Resources and Safety Engineering, Central South University, Changsha 410083, China

Received 13 February 2005; received in revised form 26 May 2005
Available online 27 July 2005

Abstract

This is the second paper of our work on structural reliability analysis for implicit performance function. The first paper proposed structural reliability analysis methods using multilayer perceptron artificial neural network [Deng, J., Gu, D.S., Li, X.B., Yue, Z.Q., 2005. Structural reliability analysis for implicit performance function using artificial neural network. Structural Safety 25 (1), 25–48]. This paper presents three radial basis function network (RBF) based reliability analysis methods, i.e. RBF based MCS, RBF based FORM, and RBF based SORM. In these methods, radial basis function network technique is adopted to model and approximate the implicit performance functions or partial derivatives. The RBF technique uses a small set of the actual data of the implicit performance functions, which are obtained via physical experiments or normal numerical analysis such as finite element methods for the complicated structural system, and are used to develop a trained RBF generalization algorithm. Then a large number of the function values and partial derivatives of implicit performance functions can be readily obtained by simply extracting information from the established and successfully trained RBF network. These function values and derivatives are used in conventional MCS, FORM or SORM to constitute RBF based reliability analysis algorithms. Examples are presented in the paper to illustrate how the proposed RBF based methods are used in structural reliability analysis. The results are well compared with those obtained by the conventional reliability methods such as the Monte-Carlo simulation, multilayer perceptrons networks, the response surface method, the FORM method 2, and so on. The examples showed the proposed approach is applicable to structural reliability analysis involving implicit performance functions.

Keywords: Radial basis function network; Implicit performance function; Reliability analysis; Monte-Carlo simulation; First-order reliability method; Second-order reliability method

* Tel.: +86 731 8836735; fax: +86 731 8879612.
E-mail address: jiandeng@csu.edu.cn

0020-7683/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.ijsolstr.2005.05.055

1. Introduction

Structural reliability analysis deals with the statistical nature of many basic variables in structural safety analysis and design. Freudenthal (1947) was among the first in the world to develop structural reliability that is the application of probabilistic methods to evaluate the safety of structures that are made of various materials. His work on the classical theory of structural reliability was summarized in a comprehensive manner by Freudenthal et al. (1966). In recent years, reliability analysis has been applied to structural design and safety reassessment of the existing structures (Ang and Tang, 1975, 1984; Ditlevsen and Madsen, 1996; Grigoriu, 2000; Harr, 1987; Madsen et al., 1986; Melchers, 1999; Nowak and Collins, 2000; Rahman and Rao, 2001). The Monte-Carlo simulation (MCS), the first-order reliability (FORM) and the second-order reliability methods (SORM) are the three methods that have been widely used to estimate the failure probability of structural systems. The MCS requires the calculations of hundreds and thousands of performance function values. The FORM and SORM generally demand the values and partial derivatives of the performance function with respect to the design random variables. Such calculations can be performed efficiently when the performance function $g(X)$ can be expressed as an explicit form or simple analytical form in terms of the basic variables X. When the performance functions are implicit, however, such calculations require additional effort and will be time-consuming. Such implicit performance functions will normally occur when costly physical experiments or computationally intensive numerical analyses such as 3-D finite element methods have to be adopted for the mechanical analysis of a structural system.

A few approaches have been developed to cope with the issues with implicit performance functions. One of the popular approaches is the response surface method (Bucher and Bourgund, 1990; Faravelli, 1989; Guan and Melchers, 1997; Rajashekhar and Ellingwood, 1993; Wong, 1985). A polynomial function is used to approximate the unknown implicit performance function. A fairly accurate estimate of the failure probability could be obtained if the selected polynomial function fits the actual limit state well. However, response surface method becomes computationally impractical for problems involving a large number of nonlinear random variables, particularly when mixed or statistically dependent random variables are involved. Besides, there is no guarantee that the fitted surface is in fact a sufficiently close fit in all regions of interest and it is difficult to construct the appropriate response surface without knowing the location of the design point (Der Kiureghian, 1996). Other modified approaches are the multi-plane surfaces method and the multi-tangent-plane surface method (Guan and Melchers, 1997), and the improved sequential response surface method (Kim and Na, 1997). These approaches can improve the accuracy of solutions obtained from the FOSM method and take less computational time than the MCS method. Yet, they are only suitable for the cases of a nonlinear concave or convex limit state surface. Bauer and Pula (2000) have also pointed out that the response surface method can sometimes lead to false design points.

The conventional MCS can also be used for the implicit performance function. However, this method was notorious for its unendurable computational cost. Some variance reduction techniques such as the importance sampling, the Latin hypercube sampling, the radial importance sampling (Melchers, 1990), and the directional importance sampling (Ditlevsen and Madsen, 1996), have been proposed to reduce the number of samples in the conventional MCS. These techniques can shorten the computational time to a certain extent. Point-estimate methods (Rosenblueth, 1975, 1981; Harr, 1987; Christian and Baecher, 2002) can facilitate reliability analyses even when performance functional relations are given as graphs or tables, rather than as mathematical functions. However, Rosenblueth's method should be based on an assumption that uncertainty can be adequately described using lower moments and correlation coefficients.

The combination of sensitivity analysis and FORM (or SORM) for implicit performance functions was discussed in detail by Haldar and Mahadevan (2000). Three methods of sensitivity analysis were presented: the finite difference, the classic perturbation, and the iterative perturbation. In the finite difference approach, it is necessary to repeat the deterministic analysis and the results would be accurate only when the input variables have small variability, and frequently some troubles arise when this method is used

(Gomes and Awruch, 2004). The classic perturbation approach can be used for problems involving modification of finite element codes. The iterative perturbation approach is suitable in the context of nonlinear structural analysis, in which an iterative process is required for mechanical response solutions. Adopting this approach involves considerable programming work and complicate computation.

During the last two decades, artificial neural network (ANN) algorithms have been rapidly developed for universal function approximator (Anjum et al., 1997; Cardaliaguet and Euvrand, 1992; Chapman and Crossland, 1995; Gomes and Awruch, 2004; Hornik et al., 1989, 1990; Schueremans, 2005; Schueremans and Van Gemert, 2005). ANN is a computational mechanism that is able to "acquire, represent, and compute a mapping from multivariate space of information to another, given a set of data representing that mapping" (Garrett, 1994). ANN is capable of learning from training examples and finding meaningful solutions without the need to specify the relationship among variables. It can capture nonlinear and complex interactions among variables in a system (Goh and Kulhawy, 2003; Masters, 1993). A multilayer perceptrons (MLP) network was developed as an approximate limit state function (Schueremans and Van Gemert, 2005; Schueremans, 2005; Shao and Murotso, 1997; Sasaki, 2001). Goh and Kulhawy (2003) used MLP approach to model the limit state surface for reliability analysis. Deng et al. (2005) described why and how to employ MLP technique to approximate the implicit performance functions and derivatives in FORM, SORM and MCS reliability analysis. Artificial neural networks in those works were almost multilayer perceptrons.

Another neural network—radial basis function network (RBF) is increasingly attracting attention recently (Chen and Chen, 1995; Haykin, 1999; Li, 1996; Mai-Duy and Tran-Cong, 2003; McDonald et al., 2000; Park and Sandberg, 1993; Warnes et al., 1998). The reason is that training of a RBF network can be essentially faster than the methods used to train MLP networks (Moody and Darken, 1989). Furthermore, the multilayer perceptron network trained with backpropagation does not yield the approximating capabilities of RBF networks. Therefore the theory of RBF neural networks is still the subject of extensive ongoing researches (Orr, 1999). Franke (1982) found radial basis functions to be superior to thin plate splines, cubic splines and B-splines, and several others. Li (1996) proved the fact that any multivariate function and all its existing derivatives can be simultaneously approximated by a radial basis function network, where the assumptions on the functions are relatively mild. Hussain et al. (2002) applied radial basis function and polynomial metamodels (i.e. response surface metamodel) to approximate the input-output functions and compared their effectiveness qualitatively and quantitatively and made conclusions that radial function metamodels provided a better fit than the polynomial metamodels. Mai-Duy and Tran-Cong (2003) presented a numerical approach, based on radial basis function networks, for the approximation of a function and its derivatives (scattered data interpolation). One remarkable feature of RBF networks is described as follows (Bishop, 1995):"RBF networks possess the property of *best approximation*. An approximation scheme has this property if, in the set of approximating functions (i.e. the set of functions corresponding to all possible choices of the adjustable parameters) there is one function which has minimum approximating error for any given function to be approximated. This property is not shared by MLP's."

Although RBF network founds some applications in deterministic engineering problems (Meckesheimer, 2001), reports on its application to a structural reliability problem are not found until recently. RBF network was used in probabilistic mechanics as only a substitution of finite element solver (Hurtado, 2002). Gomes and Awruch (2004) compared response surface method and ANN (MLP and RBF) with other alternatives to evaluate structural reliability, in which ANN was used to approximate the performance functions. In this paper, our previous researches in Deng et al. (2003, 2005) are extended to radial basis function network. RBF networks were constructed to approximate the implicit performance function (or limit state function) and the first and second order derivatives with the least efforts and without any loss of the accuracy.

The main objective of this paper is to propose three RBF network methods to compute the performance function derivatives, and then to combine them with conventional MCS, FORM and SORM and propose

three RBF reliability analysis methods: RBF based MCS, RBF based FORM and RBF based SORM. We illustrate how this RBF based approach can be used in combination with conventional reliability methods such as FORM, SORM and MCS in the facilitation of the computation of values of implicit performance function and its derivatives for reduction of the computation efforts. To investigate the suitability of these approaches, the results of the new approaches are compared with those obtained by conventional reliability methods such as the direct Monte-Carlo simulation, multilayer perceptrons, the response surface method and the FORM method 2, and so on. In Section 2, RBF network is briefly reviewed and partial derivatives of the performance function are computed using three methods of RBF: RBF network method 1, RBF network method 2, and RBF network method 3. In Section 3, three RBF based reliability analysis methods are proposed, i.e. RBF based MCS, RBF based FORM, and RBF based SORM. These methods are illustrated with the aid of four examples in Section 4. Section 5 concludes the paper.

2. Computation derivatives using RBF networks

We are concerned with the reliability analysis of multi-parameter scientific problem or engineering structures. The performance function $g(X)$, $X \in R^n$, is assumed to be implicit and evaluated experimentally or through a computationally intensive numerical simulation. It is assumed that the results of experiments or numerical simulations have yielded a set of s data points giving performance function evaluations $y_i = g(X_i)$, $i = 1, \ldots, s$, where $X_i = [x_{1i}, \ldots, x_{ni}]^T$ and $[\cdot]^T$ denotes transpose operation, n is the number of dimensions, s is the number of input sample pairs. The s data samples pairs (X_i, y_i) $(i = 1, \ldots, s)$ are used as the training samples of RBF networks.

2.1. RBF network

The RBF network has a feed-forward structure consisting of a single hidden layer of locally tuned units which are fully interconnected to an output layer of linear units as shown in Fig. 1.

All hidden units simultaneously receive the n-dimensional real-valued input vector $\mathbf{x}$. Each hidden unit output $\phi_i(\mathbf{x})$ $(i = 1, \ldots, m)$ is obtained by calculating the "closeness" of the input $\mathbf{x}$ to an n-dimensional

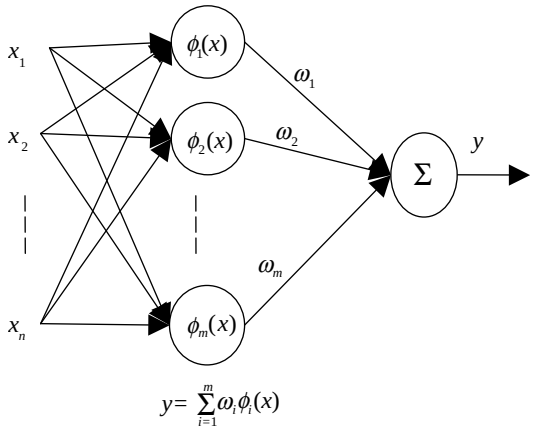

Fig. 1. A radial basis function neural network consisting of a single hidden layer of locally tuned units which are fully interconnected to an output layer of linear units.

parameter vector, where i is associated with the ith hidden unit, m is the number of radial basis functions, i.e. the number of hidden units, and $m \leqslant s$, s is the number of input pairs. Here, the response characteristics of the ith hidden unit are given by

$$\phi^{(i)}(r) = \phi^{(i)}\left(\left|x - c^{(i)}\right|\right) \quad i = 1, \ldots, m \tag{2.1}$$

where $\phi^{(i)}(r)$ is called radial basis function; $r = \|x - c^{(i)}\|$ denotes the Euclidean distance, and $r = \sqrt{(x - c^{(i)}) \cdot (x - c^{(i)})} = \sqrt{\sum_{j=1}^{n} [x_j - c_j^{(i)}]^2}$; $c^{(i)}$ is the centroid of the radial basis function that can be chosen from the data points.

The most commonly used radial basis functions are multiquadrics, inverse multiquadrics, Gaussians, thin plate spline and cubic, which are listed in the second column of Table 1. Given an input vector $\mathbf{x}$, the output of the RBF network is given by

$$y = f(x) = \sum_{i=1}^{m} w^{(i)} \phi^{(i)}(x) \tag{2.2}$$

Before the establishment of a RBF model, training sets or samples should be created. Perform the calculation for $g(X)$ so as to cover the range of values of $\mathbf{x}$ which are likely to occur. For some problems, these calculations may involve procedure such as finite element method. The number of sampling points required to accurately model the performance function is dependent on the number of random variables, the nonlinearity of the problem considered and the assigned computation accuracy. Guidelines on the "design" of sampling points can be found in various statistical textbooks (Lawson and Erjavee, 2001) and neural network monographs (Hecht-Nielsen, 1989; Haykin, 1999). These sampling data points are then used as the training and testing data in the RBF computations so as to approximately represent the performance function $g(X)$.

Consider a training set of s labeled pairs (X_i, y_i), $i = 1, \ldots, s$, which represent samples of a multivariate implicit performance function. Efficient training algorithms have been developed to minimize the sum squared error by adaptively updating the free parameters of the RBF network. These parameters are the receptive field *centers* (Centroids) of the hidden layer, the receptive field *widths*, and the output layer *weights*. Finding the centers, width and weights of the hidden nodes constitutes the training of an RBF network. For optimal performance of an RBF network the position of the centers and width of the hidden nodes is critical. Generally, the position of the centroids and the width of the radial basis functions are obtained by an unsupervised learning rule, whereas the weights of the output layer are calculated by a supervised, single-shot process using pseudo-inverse matrices, normal equations method or singular value decomposition (Press et al., 1992). According to Eq. (2.2), the RBF network may be viewed as approximating a desired function $f(x)$ by superposition of non-orthogonal bell-shaped basis functions. The degree of accuracy can be controlled by the above-mentioned three parameters: the centers, width and weights. For more details on the training algorithm of RBF network, refer to Bishop (1995) and Haykin (1999), etc.

An example is presented to demonstrate the capability of the RBF network to map (approximate) a complicated nonlinear performance function. The performance function has the following form

$$y = f(x_1, x_2, x_3, x_4) = x_1 x_2 + (x_3)^2 - (x_4)^{1/4} \tag{2.3}$$

Series of input patterns were randomly generated for training and testing the RBF network using the program described in Goh (1994). The training and testing data presented in Table 2 consist of 20 patterns and 10 patterns, respectively. Multiquadrics RBFs are used. The width constant of RBF is 8, and the number of RBF in the hidden layer is 13. Comparative study is shown in Table 2 on model capability of MLP,

Table 1
Classical RBF functions and derivatives

Classical RBF name	Classical RBF $\phi^{(i)}(r)$	First order derivatives $h^{(i)}(x)=\frac{\partial\phi^{(i)}}{\partial x_j}$	Second order derivatives $\overline{h}_j^{(i)}(x)=\frac{\partial h^{(i)}}{\partial x_j}$	Second order derivatives $\overline{h}_k^{(i)}(x)=\frac{\partial h^{(i)}}{\partial x_k}$
Multi-quadrics	$\sqrt{r^2+a^{(i)2}}$	$\frac{x_j-c_j^{(i)}}{(r^2+a^{(i)2})^{0.5}}$	$\frac{r^2+a^{(i)2}-\left(x_j-c_j^{(i)}\right)^2}{(r^2+a^{(i)2})^{1.5}}$	$-\frac{\left(x_j-c_j^{(i)}\right)\left(x_k-c_k^{(i)}\right)}{(r^2+a^{(i)2})^{1.5}}$
Inverse multi-quadrics	$\frac{1}{\sqrt{r^2+a^{(i)2}}}$	$-\frac{x_j-c_j^{(i)}}{(r^2+a^{(i)2})^{1.5}}$	$\frac{3\left(x_j-c_j^{(i)}\right)^2}{(r^2+a^{(i)2})^{2.5}}-\frac{1}{(r^2+a^{(i)2})^{1.5}}$	$\frac{3\left(x_j-c_j^{(i)}\right)\left(x_k-c_k^{(i)}\right)}{(r^2+a^{(i)2})^{2.5}}$
Gaussians	$\exp\left(-\frac{r^2}{a^{(i)2}}\right)$	$\frac{-2\left(x_j-c_j^{(i)}\right)}{a^{(i)2}}\exp\left(-\frac{r^2}{a^{(i)2}}\right)$	$\frac{2}{a^{(i)2}}\left[\frac{2}{a^{(i)2}}\left(x_j-c_j^{(i)}\right)^2-1\right]\exp\left(-\frac{r^2}{a^{(i)2}}\right)$	$\frac{4\left(x_j-c_j^{(i)}\right)\left(x_k-c_k^{(i)}\right)}{a^{(i)4}}\exp\left(-\frac{r^2}{a^{(i)2}}\right)$
Thin plate spline	$r^2\ln(a^{(i)r})$	$\left(x_j-c_j^{(i)}\right)[2\ln(a^{(i)}r)+1]$	$\ln(a^{(i)}r)+\frac{2\left(x_j-c_j^{(i)}\right)^2}{r^2}+1$	$\frac{2\left(x_j-c_j^{(i)}\right)\left(x_k-c_k^{(i)}\right)}{r^2}$
Cubic	$\left(r+a^{(i)}\right)^3$	$\frac{3\left(r+a^{(i)}\right)^2\left(x_j-c_j^{(i)}\right)}{r}$	$\frac{3}{r}\left[\left(r+a^{(i)}\right)^2+2\left(1+\frac{a^{(i)}}{r}\right)\left(x_j-c_j^{(i)}\right)^2\right]-\frac{3}{r^3}\left[\left(r+a^{(i)}\right)^2\left(x_j-c_j^{(i)}\right)^2\right]$	$\frac{3\left(x_j-c_j^{(i)}\right)\left(x_k-c_k^{(i)}\right)\left(r+a^{(i)}\right)\left(1-\frac{a^{(i)}}{r}\right)}{r^2}$

Note: $r=\left\|x-c^{(i)}\right\|=\sqrt{(x-c^{(i)})\cdot(x-c^{(i)})}=\sqrt{\sum_{j=1}^{n}[x_j-c_j^{(i)}]^2}$; $a^{(i)}$ is referred to as the width of the ith radial basis function.

Table 2
Comparison of model capability of MLP, RBF and polynomial regression

x_1	x_2	x_3	x_4	Actual y	Computed y (MLP)	Computed y (polynomial regression)	Computed y (RBF network)
Training data							
2.5	4.6	3.7	1.3	24.12	16.07	24.00	24.06
2.4	3.5	2.1	3	11.49	11.28	12.33	11.43
2.9	2	1.5	1	7.05	13.74	6.89	7.02
3.3	2.4	4.2	3	24.24	21.99	25.08	24.15
2.2	5	2.2	3.7	14.45	12.91	16.01	14.52
4	4.1	1.8	1.2	18.59	29.32	18.17	18.63
1.1	2	1.2	4.3	2.20	24.17	0.99	2.23
1.8	1.8	4.9	1.9	26.08	10.57	25.33	26.03
4	4.7	2.1	4.6	21.75	13.76	20.60	21.89
4.1	4.6	3.1	3.3	27.12	20.70	25.57	27.03
4.1	4.8	2.6	2.4	25.20	24.89	23.35	24.84
1.6	2.1	4.1	3.6	18.79	19.26	16.82	18.75
2.2	5	3.1	1.4	19.52	20.25	19.60	19.60
2.4	4.5	3.9	4.1	24.59	25.62	25.59	24.58
4.2	3.8	3.7	3.5	28.28	28.48	27.99	28.33
4.8	2.2	1.7	4.2	12.02	10.94	14.48	12.00
4.3	1.1	2.8	2.4	11.33	12.83	11.96	11.15
1.2	3	4.5	2.7	22.57	23.01	22.98	22.65
1.6	4.7	4.2	1.5	24.05	23.80	25.86	24.04
4.3	2.7	3.2	1.8	20.69	21.75	20.54	21.19
				MSE	0.77	1.41	0.024
Testing data							
3.3	1.5	3.5	1.5	16.09	16.07	14.57	16.76
2.9	2.4	2.5	3.8	11.81	11.28	11.39	11.98
2.5	3	2.6	3.7	12.87	13.74	12.63	13.20
5	2.5	3.3	1.6	22.27	21.99	22.44	22.54
2.8	1.2	3.4	2.5	13.66	12.91	11.29	13.81
4.3	4.4	3.7	2.1	31.41	29.32	29.89	30.61
5	2.4	3.7	4.6	24.23	24.17	25.07	24.25
4.2	2.2	1.8	1.6	11.36	10.57	12.63	11.40
3.4	4	1.7	3.5	15.12	13.76	16.55	14.88
2.5	1.5	4.3	3.2	20.90	20.70	19.77	20.55
				MSE	0.86	1.62	0.15

RBF network and polynomial regression. The results of MLP and polynomial regression are from Goh and Kulhawy (2003). Table 2 shows that the mean squared error (MSE) for the polynomial regression model is about two times that of the MLP network, and about ten times that of the RBF network.

2.2. Computation derivatives using RBF network

After the RBF model is established and trained satisfactorily, partial derivatives of performance functions can be computed by extracting the information (centers, width and weights etc.) of the trained RBF model. In this process, mathematical expressions are obtained that approximately represent the implicit performance function and the partial derivatives. Since there exist differential rule and integral rule in calculus, three methods are reported here to compute performance function derivatives by RBF network. Differential rule is used in RBF network method 1, while integral rule is used in RBF network method 2 and RBF network method 3, in which the approximation of two-variate function $f(x_1, x_2)$ is considered,

and the procedure for functions of three or more variables can be similarly developed (Mai-Duy and Tran-Cong, 2003).

2.2.1. RBF network method 1

In this method, the original function $f(x)$ is first approximated in terms of radial basis functions as indicated in Eq. (2.2). The partial derivatives are calculated by the differential rule.

According to Eq. (2.2), the first order partial derivatives of the approximate function $f(x)$ can be calculated as follows

$$f_{,j}(x) = \frac{\partial f(x)}{\partial x_j} = \sum_{i=1}^{m} w^{(i)} h^{(i)}(x) = \sum_{i=1}^{m} w^{(i)} \frac{\partial \phi^{(i)}(x)}{\partial x_j} \tag{2.4}$$

where $f_{,j}(x)$ is the derivative function with respect to x_j. $h^{(i)}(x) = \frac{\partial \phi^{(i)}(x)}{\partial x_j}$ is the corresponding basis function for the derivative function $f_{,j}(x)$ which is obtained by differentiating the original basis function $\phi^{(i)}(x)$ which is continuously differentiable. The $h^{(i)}(x)$ of the most commonly used radial basis functions is listed in the third column of Table 1.

Similarly, the second order partial derivatives of the approximate function $f(x)$ can be calculated as follows

$$f_{,jj}(x) = \frac{\partial^2 f(x)}{\partial x_j \partial x_j} = \sum_{i=1}^{m} w^{(i)} \overline{h}_j^{(i)}(x) = \sum_{i=1}^{m} w^{(i)} \frac{\partial h^{(i)}(x)}{\partial x_j} = \sum_{i=1}^{m} w^{(i)} \frac{\partial^2 \phi^{(i)}(x)}{\partial^2 x_j} \tag{2.5}$$

$$f_{,jk}(x) = \frac{\partial^2 f(x)}{\partial x_j \partial x_k} = \sum_{i=1}^{m} w^{(i)} \overline{h}_k^{(i)}(x) = \sum_{i=1}^{m} w^{(i)} \frac{\partial h^{(i)}(x)}{\partial x_k} = \sum_{i=1}^{m} w^{(i)} \frac{\partial^2 \phi^{(i)}(x)}{\partial x_j \partial x_k} \tag{2.6}$$

The $\overline{h}_j^{(i)}(x)$ and $\overline{h}_k^{(i)}(x)$ of the most commonly used radial basis functions are listed in the fourth and fifth column of Table 1. From Eqs. (2.4)–(2.6), it can be seen that computation of derivatives is intrinsically to extract information from the established and successfully trained RBF network.

2.2.2. RBF network method 2

In this method, the first order partial derivative of $f(x_1,x_2)$ with respect to x_1, denoted by $f_{,1}$, is first approximated in terms of radial basis functions. The original function $f(x_1,x_2)$ is calculated by the integral rule.

$$f_{,1}(x_1, x_2) = \sum_{i=1}^{m} w^{(i)} \phi^{(i)}(x_1, x_2) \tag{2.7}$$

where $\{\phi^{(i)}(x_1, x_2)\}_{i=1}^{m}$ is a set of radial basis functions and $\{w^{(i)}\}_{i=1}^{m}$ is a set of corresponding weights.

The original function can be calculated by integration method as follows

$$\begin{aligned} f(x_1, x_2) &= \int f_{,1}(x_1, x_2)\, \mathrm{d}x_1 = \int \sum_{i=1}^{m} w^{(i)} \phi^{(i)}(x_1, x_2)\, \mathrm{d}x_1 = \sum_{i=1}^{m} w^{(i)} \int \phi^{(i)}(x_1, x_2)\, \mathrm{d}x_1 \\ &= \sum_{i=1}^{m} w^{(i)} H^{(i)}(x_1, x_2) + C_1(x_2) \end{aligned} \tag{2.8}$$

where $\{H^{(i)}(x_1, x_2)\}_{i=1}^{m}$ is the set of corresponding radial basis functions for the original function and is given below. Gaussian radial basis function can't be obtained analytically.

(1) For multiquadrics

$$H^{(i)}(x_1, x_2) = \frac{\left(x_1 - c_1^{(i)}\right)\sqrt{r^2 + a^{(i)2}}}{2} + \frac{r^2 - \left(x_1 - c_1^{(i)}\right)^2 + a^{(i)2}}{2} \ln\left(\left(x_1 - c_1^{(i)}\right) + \sqrt{r^2 + a^{(i)2}}\right) \tag{2.9}$$

(2) For inverse multiquadrics

$$H^{(i)}(x_1,x_2)=\ln\left(\left(x_1-c_1^{(i)}\right)+\sqrt{r+a^{(i)}}\right) \tag{2.10}$$

(3) For thin plate spline

$$\begin{aligned}H^{(i)}(x_1,x_2)=&\frac{1}{3}\left(x_1-c_1^{(i)}\right)^3\ln(a^{(i)}r^2)-\frac{2}{9a^{(i)}}\left(x_1-c_1^{(i)}\right)^3+\left(x_1-c_1^{(i)}\right)\left(x_2-c_2^{(i)}\right)^2\ln(a^{(i)}r^2)\\&-\frac{4}{3a^{(i)}}\left(x_1-c_1^{(i)}\right)\left(x_2-c_2^{(i)}\right)^2+\frac{4}{3a^{(i)}}\left(x_2-c_2^{(i)}\right)\arctan\left(\frac{x_1-c_1^{(i)}}{x_2-c_2^{(i)}}\right)\end{aligned} \tag{2.11}$$

$C_1(x_2)$ in Eq. (2.8) is a function of the variables x_2 and can be interpolated as follows.

$$C_1''(x_2)=\sum_{i=1}^{M}\overline{w}^{(i)}g^{(i)}(x_2) \tag{2.12}$$

$$C_1'(x_2)=\int C_1''(x_2)\,\mathrm{d}x_2=\sum_{i=1}^{M}\overline{w}^{(i)}H^{(i)}(x_2)+\widehat{C}_1 \tag{2.13}$$

$$C_1(x_2)=\int C_1'(x_2)\,\mathrm{d}x_2=\sum_{i=1}^{M}\overline{w}^{(i)}\overline{H}^{(i)}(x_2)+\widehat{C}_1x_2+\widehat{C}_2 \tag{2.14}$$

where $\widehat{C}_1$ and $\widehat{C}_2$ are constants of integration; $\overline{w}^{(i)}$ is the corresponding weights; and M is the number of centers whose x_2 coordinates are distinct.

(1) For multiquadrics

$$H^{(i)}(x_2)=\frac{\left(x_2-c_2^{(i)}\right)\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}}{2}+\frac{a^{(i)2}}{2}\ln\left(\left(x_2-c_2^{(i)}\right)+\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\right) \tag{2.15}$$

$$\begin{aligned}\overline{H}^{(i)}(x_2)=&\int H^{(i)}(x_2)\,\mathrm{d}x_2\\=&\frac{\left(\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}\right)^{1.5}}{6}+\frac{a^{(i)2}}{2}\left(x_2-c_2^{(i)}\right)\ln\left(\left(x_2-c_2^{(i)}\right)+\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\right)\\&-\frac{a^{(i)2}}{2}\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\end{aligned} \tag{2.16}$$

(2) For inverse multiquadrics

$$H^{(i)}(x_2)=\ln\left(\left(x_2-c_2^{(i)}\right)+\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\right) \tag{2.17}$$

$$\begin{aligned}\overline{H}^{(i)}(x_2)=&\int H^{(i)}(x_2)\,\mathrm{d}x_2=\left(x_2-c_2^{(i)}\right)\ln\left(\left(x_2-c_2^{(i)}\right)+\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\right)\\&-\sqrt{\left(x_2-c_2^{(i)}\right)^2+a^{(i)2}}\end{aligned} \tag{2.18}$$

The training to determine the weights in Eqs. (2.7) and (2.8) is equivalent to minimization of the following sum squared error (SSE) (Mai-Duy and Tran-Cong, 2003)

$$\mathrm{SSE} = \sum_{j=1}^{n} \left[y^{(j)} - f\left(x_1^{(j)}, x_2^{(j)}\right) \right]^2 \tag{2.19}$$

Eq. (2.8) is used in (2.19) in the minimization procedure, which results in a system of equations in terms of the unknown parameters, which are composed of the weights in (2.8), the second set of weights in (2.14) and the constants of integration $\widehat{C}_1$, $\widehat{C}_2$. The data used in training the network for the derivatives and the performance functions just consists of a set of discrete values $y_i = g(X_i)$, $i = 1, \ldots, s$ of the dependent variables. Upon applying the general linear least squares principle, a system of linear algebraic equations containing unknown variables can be obtained. Singular value decomposition method can be used to solve Eq. (2.19) for the unknowns and the constant of integration in the remainder of this paper. After solving Eq. (2.19), a set of weights can be obtained and used for approximating the derivative function via Eq. (2.7) and together with $\overline{w}^{(i)}$, $\widehat{C}_1$ and $\widehat{C}_2$ for estimating the original function via Eq. (2.8).

The strategy of approximation is similar for the derivative function of $f(x_1, x_2)$ with respect to the variable x_2. Once the first order derivatives are obtained, the second order partial derivative of $f(x_1, x_2)$ with respect to x_1 is first approximated in terms of radial basis functions, the first order derivatives function can be calculated by integration method, and then the second order partial derivative can be solved.

2.2.3. RBF network method 3

In this method, the second order derivative functions are first approximated in terms of radial basis functions. The first order derivative is calculated by the integral rule. And the original function is calculated by integrating the first order derivative.

There are two cases: in one case, $f_{,11}$ $(\frac{\partial^2 f}{\partial x_1^2})$ or $f_{,22}$ $(\frac{\partial^2 f}{\partial x_2^2})$ is first approximated; in the other case, $f_{,12}$ $(\frac{\partial^2 f}{\partial x_1 x_2})$ or $f_{,21}$ $(\frac{\partial^2 f}{\partial x_2 x_1})$ is first approximated.

2.2.3.1. In the case of $f_{,11}$. Suppose $f_{,11}$ is first approximated in terms of radial basis functions

$$f_{,11}(x_1, x_2) = \sum_{i=1}^{m} w^{(i)} \phi^{(i)}(x_1, x_2) \tag{2.20}$$

where $\{\phi^{(i)}(x_1, x_2)\}_{i=1}^{m}$ is a set of radial basis functions and $\{w^{(i)}\}_{i=1}^{m}$ is the set of corresponding weights. The first derivative function, $f_{,1}$, can be calculated as follows

$$\begin{aligned} f_{,1}(x_1, x_2) &= \int f_{,11}(x_1, x_2)\, \mathrm{d}x_1 = \int \sum_{i=1}^{m} w^{(i)} \phi^{(i)}(x_1, x_2)\, \mathrm{d}x_1 = \sum_{i=1}^{m} w^{(i)} \int \phi^{(i)}(x_1, x_2)\, \mathrm{d}x_1 \\ &= \sum_{i=1}^{m} w^{(i)} H^{(i)}(x_1, x_2) + C_1(x_2) \end{aligned} \tag{2.21}$$

The basis functions for $f_{,1}$ are given by (2.9) or (2.10). $C_1(x_2)$ can be calculated by Eq. (2.14).

The original function, f, can be calculated as follows

$$\begin{aligned} f(x_1, x_2) &= \int f_{,1}(x_1, x_2)\, \mathrm{d}x_1 = \int \left[\sum_{i=1}^{m} w^{(i)} H^{(i)}(x_1, x_2) + C_1(x_2) \right] \mathrm{d}x_1 \\ &= \sum_{i=1}^{m} \left[w^{(i)} \int H^{(i)}(x_1, x_2)\, \mathrm{d}x_1 \right] + \int C_1(x_2)\, \mathrm{d}x_1 \\ &= \sum_{i=1}^{m} w^{(i)} \overline{H}^{(i)}(x_1, x_2) + \int C_1(x_2)\, \mathrm{d}x_1 + C_2(x_2) \end{aligned} \tag{2.22}$$

The basis functions for f are obtained by integrating (2.9) or (2.10) and shown below.

(1) For multiquadrics

$$\begin{aligned}\overline{H}^{(i)}(x_1,x_2) &= \int H^{(i)}(x_1,x_2)\,\mathrm{d}x_1 \\ &= \frac{(r^2+a^{(i)2})^{1.5}}{6} + \frac{r^2-\left(x_1-c_1^{(i)}\right)^2+a^{(i)2}}{2}\left(x_1-c_1^{(i)}\right)\ln\left(\left(x_1-c_1^{(i)}\right)+\sqrt{r^2+a^{(i)2}}\right) \\ &\quad - \frac{r^2-\left(x_1-c_1^{(i)}\right)^2+a^{(i)2}}{2}\sqrt{r^2+a^{(i)2}}\end{aligned} \tag{2.23}$$

(2) For inverse multiquadrics

$$\overline{H}^{(i)}(x_1,x_2) = \int H^{(i)}(x_1,x_2)\,\mathrm{d}x_1 = \left(x_1-c_1^{(i)}\right)\ln\left(\left(x_1-c_1^{(i)}\right)+\sqrt{r^2+a^{(i)2}}\right) - \sqrt{r^2+a^{(i)2}} \tag{2.24}$$

The original function is calculated as follows

$$f(x_1,x_2) = \sum_{i=1}^{M} w^{(i)}\overline{H}^{(i)}(x_1,x_2) + C_1(x_2)x_1 + C_2(x_2) \tag{2.25}$$

where $C_1(x_2)$ and $C_2(x_2)$ are constants of integration which are interpolated in the same manner as showed by Eqs. (2.12)–(2.14). Similarly, the SSE function as in Eq. (2.19) can be established and solved using singular value decomposition method.

2.2.3.2. In the case of $f_{,21}$. Suppose $f_{,21}$ is first approximated in terms of radial basis functions

$$f_{,21}(x_1,x_2) = \sum_{i=1}^{m} w^{(i)}\phi^{(i)}(x_1,x_2) \tag{2.26}$$

where $\{\phi^{(i)}(x_1,x_2)\}_{i=1}^{m}$ is a set of radial basis functions and $\{w^{(i)}\}_{i=1}^{m}$ is the set of corresponding weights.

The first derivative function, $f_{,2}$, can be calculated as follows

$$\begin{aligned}f_{,2}(x_1,x_2) &= \int f_{,21}(x_1,x_2)\,\mathrm{d}x_1 = \int \sum_{i=1}^{m} w^{(i)}\phi^{(i)}(x_1,x_2)\,\mathrm{d}x_1 = \sum_{i=1}^{m} w^{(i)}\int \phi^{(i)}(x_1,x_2)\,\mathrm{d}x_1 \\ &= \sum_{i=1}^{m} w^{(i)}H^{(i)}(x_1,x_2) + C_1(x_2)\end{aligned} \tag{2.27}$$

The basis functions for $f_{,2}$ are given by (2.9) or (2.10). $C_1(x_2)$ can be calculated by Eq. (2.14).

The original function, f, can be calculated as follows

$$\begin{aligned}f(x_1,x_2) &= \int f_{,2}(x_1,x_2)\,\mathrm{d}x_2 = \int \left[\sum_{i=1}^{m} w^{(i)}H^{(i)}(x_1,x_2) + C_1(x_2)\right]\mathrm{d}x_2 \\ &= \sum_{i=1}^{m}\left[w^{(i)}\int H^{(i)}(x_1,x_2)\,\mathrm{d}x_2\right] + \int C_1(x_2)\,\mathrm{d}x_2\end{aligned}$$

$$= \sum_{i=1}^{m} w^{(i)} \widehat{H}^{(i)}(x_1, x_2) + \int C_1(x_2)\,\mathrm{d}x_2 + C_2(x_1)$$
$$= \sum_{i=1}^{m} w^{(i)} \widehat{H}^{(i)}(x_1, x_2) + \sum_{i=1}^{M} \overline{w}^{(i)} \widetilde{H}(x_2) + \frac{1}{2}\widehat{C}_1 x_2^2 + \widehat{C}_2 x_2 + C_3(x_1) \tag{2.28}$$

Similarly, the SSE function as in Eq. (2.19) can be established and solved using singular value decomposition method. The basis functions $\widehat{H}^{(i)}(x_1, x_2)$ are obtained by integrating (2.9) or (2.10) and are complex, so only those for inverse multiquadrics are listed in (2.29) and (2.30). The integrals listed in Appendix A are useful for integrating (2.9) or (2.10). $C_3(x_1)$ can be interpolated in the same manner as showed by (2.12)–(2.14).

For inverse multiquadrics,

$$\widetilde{H}(x_2) = \frac{\left(x_2 - c_2^{(i)}\right)^2}{2} \ln\left(\left(x_2 - c_2^{(i)}\right) + \sqrt{\left(x_2 - c_2^{(i)}\right)^2 + a^{(i)2}}\right) - \frac{3\left(x_2 - c_2^{(i)}\right)}{4}\sqrt{\left(x_2 - c_2^{(i)}\right)^2 + a^{(i)2}} \frac{a^{(i)2}}{4} \ln\left(\left(x_2 - c_2^{(i)}\right) + \sqrt{\left(x_2 - c_2^{(i)}\right)^2 + a^{(i)2}}\right); \tag{2.29}$$

$$\widehat{H}^{(i)}(x_1, x_2) = \int H^{(i)}(x_1, x_2)\,\mathrm{d}x_2 = \int \ln\left[\left(x_1 - c_1^{(i)}\right) + \sqrt{r^2 + a^{(i)2}}\right]\mathrm{d}x_2 = \left(x_2 - c_2^{(i)}\right)\ln(b + s) - h. \tag{2.30}$$

where

$$b = x_1 - c_1^{(i)}, \quad s = \left[q + \left(x_2 - c_2^{(i)}\right)^2\right]^{1/2}$$
$$h = \sqrt{q}\frac{\mathrm{e}^u - \mathrm{e}^{-u}}{2} - bu + v * \frac{b^2 - q}{\sqrt{q}}$$
$$q = b^2 + a^{(i)2}, \quad u = \ln\left(\frac{x_2 - c_2^{(i)}}{\sqrt{q}} + \sqrt{\frac{\left(x_2 - c_2^{(i)}\right)^2}{q} + 1}\right)$$
$$v = 2 * \frac{\sqrt{q}}{a^{(i)}} * \arctan\left(\frac{\mathrm{e}^u\sqrt{q} + b}{a^{(i)}}\right)$$

3. RBF based reliability analysis approaches

RBF-based reliability analysis is to construct an RBF network to approximate and replace the (implicit and often complex) performance function or the derivative functions. Then values of performance function are easily available because of the robust RBF generalization capability. The values of the first-order or the second-order partial derivatives can be readily computed as mentioned in Section 2. Because the values and partial derivatives of the performance function are both readily available through the RBF algorithms, the FORM, the SORM or the MCS can be implemented without difficulty. A strict mathematical verification on using a three-layer artificial neural network to substitute for an implicit performance function is presented in

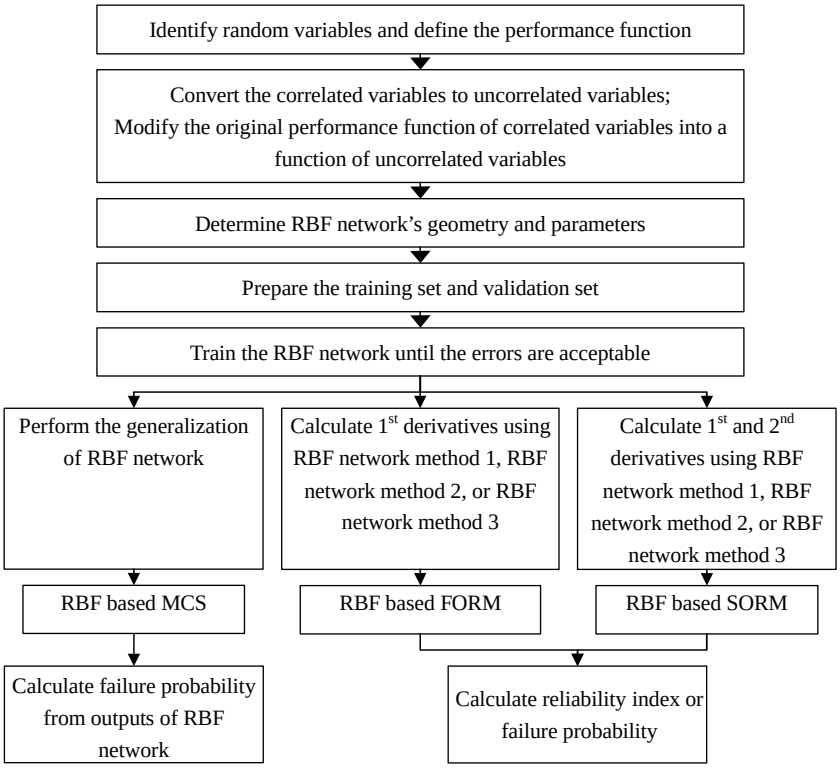

Fig. 2. Outline of RBF based MCS, RBF based FORM and RBF based SORM.

Deng et al. (2003, 2005). In the following, three RBF-based reliability analysis methods are consecutively introduced, i.e. RBF based MCS, RBF based FORM, and RBF based SORM, which are outlined in Fig. 2.

3.1. RBF based MCS

The computation procedure of the RBF based MCS is proposed in Fig. 2. The RBF is used to model or approximate the performance function, $g(X)$. This RBF based MCS employs the robust generality capability of RBF to compute the N values of implicit performance function. Among the N results, suppose there are N_f number of the performance function whose value is less than zero. Then the probability of failure p_f can be estimated using the following equation.

$$p_f = P[g(X) < 0] = \frac{N_f}{N}. \tag{3.1}$$

When some or all the random variables are correlated, the first step that needs consideration is to convert the correlated random variables to uncorrelated or statistically independent random variables. The methods proposed by Morgenstern (1956) or Nataf (1962) can be used to convert the correlated random variables to uncorrelated random variables. The second step is to modify the original performance function expressed in terms of correlated variables into a function of uncorrelated random variables. For more details read the reference (Haldar and Mahadevan, 2000). The RBF network could be established to model the modified implicit performance function, and MCS method can be performed as described in the preceding paragraph.

3.2. RBF based FORM

Outline of RBF based FORM is showed in Fig. 2. The RBF based FORM differentiates itself from other FORMs in that it employs the RBF to simultaneously approximate the performance function and its first-order partial derivatives. The performance function is supposed to be

$$Z = g(X) = g(X_1, X_2, \ldots, X_n) \tag{3.2}$$

where the vector $X = \{X_1, X_2, \ldots, X_n\}$ is the basic random variables in the original coordinate system and n is the number of random variables. The vector $X' = \{X'_1, X'_2, \ldots, X'_n\}$ is random variables in the reduced coordinate system (equivalent standard normal space) (Haldar and Mahadevan, 2000). It is denoted by the vector $\mathbf{x}^* = \{x_1^*, x_2^*, \ldots, x_n^*\}$ as the coordinates of the design point in the original coordinate system, and by $\mathbf{x}'^* = \{x_1'^*, x_2'^*, \ldots, x_n'^*\}$ as the coordinates of the design point in the reduced coordinate system. The calculation steps of RBF based FORM can be described as follows:

Step 1. Identify the random variables, specify their associated probabilistic characters (such as mean values and coefficient of variation), and define the performance function.

Step 2. Assume initial values of the design point $\mathbf{x}^* = \{x_1^*, x_2^*, \ldots, x_n^*\}$, and compute the corresponding value of the performance function $g(\cdot)$.

Step 3. Compute the mean $(\mu_{X_i}^N)$ and standard deviation $(\sigma_{X_i}^N)$ at the design point of the equivalent normal distribution for those variables that are non-normal by using the Rackwitz and Fiessler method (1976).

$$\sigma_{X_i}^N = \frac{\phi\{\Phi^{-1}[F_{X_i}(x_i^*)]\}}{f_{X_i}(x_i^*)} \tag{3.3}$$

and

$$\mu_{X_i}^N = x_i^* - \Phi^{-1}[F_{X_i}(x_i^*)]\sigma_{X_i}^N \tag{3.4}$$

where Φ^{-1} is the inverse CDF of the standard normal variate, $\mu_{X_i}^N$ and $\sigma_{X_i}^N$ are the mean and standard deviation of he equivalent normal variable at the design point, $F_{X_i}(x_i^*)$ is the CDF of the original nonnormal variables, ϕ and $f_{X_i}(x_i^*)$ are the PDFs of the equivalent standard normal and the original nonnormal random variable. The coordinates of the design point in the equivalent standard normal space are

$$x_i'^* = \frac{x_i^* - \mu_{X_i}^N}{\sigma_{X_i}^N} \tag{3.5}$$

Step 4. Establish the RBF model of the performance function. Compute the first order partial derivatives $\left.\frac{\partial g}{\partial X_i}\right|_{x_i^*}$ using the RBF technique at the design point x_i^*.

Step 5. Compute the partial derivatives $\left.\frac{\partial g}{\partial X'_i}\right|_{x_i'^*}$ in the equivalent standard normal space by the chain rule of differentiation.

$$\left.\frac{\partial g}{\partial X'_i}\right|_{x_i'^*} = \left(\left.\frac{\partial g}{\partial X_i}\right|_{x_i^*}\right) \cdot \sigma_{X_i}^N \tag{3.6}$$

Step 6. Compute the new values for the design point in the equivalent standard normal space ($\mathbf{x}'^*$) using the following recursive formula:

$$\mathbf{x}_{k+1}'^* = \frac{1}{|\nabla g(\mathbf{x}_k'^*)|^2}\left[\nabla g(\mathbf{x}_k'^*)^t \mathbf{x}_k'^* - g(\mathbf{x}_k^*)\right]\nabla g(\mathbf{x}_k'^*) \tag{3.7}$$

where $g(\mathbf{x}_k'^*)$ and $\nabla g(\mathbf{x}_k'^*)$ are respectively the values and the gradient vector of the performance function at point $\mathbf{x}_k'^*$ at the $k-$th iteration point; Subscript k refers to the iteration number. Therefore $\mathbf{x}_k'^*$ is a vector with the components $\{x_{1k}'^*, x_{2k}'^*, \ldots, x_{nk}'^*\}^T$; $\mathbf{x}_{k+1}'^*$ is the vector at the $(k+1)$-th iteration point.

Step 7. Compute the distance β to this new design point from the origin and check the convergence criterion for $\beta(|\Delta\beta| \leqslant \varepsilon_1\ ?)$.

$$\beta = \sqrt{\sum_{i=1}^{n} (x_i'^*)^2} \tag{3.8}$$

where ε_1 is a predetermined tolerance level, say 0.001.

Step 8. Compute the new values for the design point in the original space (x_i^*) as follows

$$x_i^* = \mu_{X_i}^N + \sigma_{X_i}^N x_i'^*. \tag{3.9}$$

Compute the value of the performance function $g(\cdot)$ for this new design point, and check the convergence criterion for $g(\cdot)[|g(\cdot)| \leqslant \varepsilon_2\ ?]$, where ε_2 is a predetermined tolerance level, say 0.001. If both convergence criterions are satisfied, stop. Otherwise, repeat steps 3 through 8 until convergence occurs.

3.3. RBF based SORM

Outline of RBF based SORM is showed in Fig. 2. The failure probability p_f can be calculated using the second-order reliability method (SORM) as follows (Breitung, 1984).

$$p_f = \Phi(-\beta) \prod_{i=1}^{n-1} (1 + \beta k_i)^{-1/2} \tag{3.10}$$

where $\Phi(\cdot)$ is the CDF of the standard normal variate, β is the reliability index using FORM in Section 3.3, and k_i is the principal curvature of the limit state at the minimum distance point, n is the number of basic variables. Eq. (3.10) shows that SORM improves the FORM result by including additional information about the curvature of the limit state.

In addition to presenting the results of the probabilistic reliability methods in terms of probability of failure, the reliability index β_2 is commonly used in SORM analyses, as was proposed by Hasofer and Lind (1974). The relation between the probability of failure p_f and the reliability index β_2 is given by

$$p_f = \Phi(-\beta_2) = 1 - \Phi(\beta_2) \tag{3.11}$$

where $\Phi(\cdot)$ is the CDF of the standard normal variate, β_2 is the reliability index computed by using SORM.

The RBF based SORM differentiates itself from other SORMs (Breitung, 1984; Nowak and Collins, 2000; Rackwitz and Fiessler, 1978) in that it employs the RBF to compute the first and second-order derivatives of the implicit performance function. Eq. (3.10) indicates that to compute failure probability of SORM, the reliability index using FORM β and the principal curvature k_i should be known first. The computation steps of the principal curvature k_i are outlined as follows.

Step 1. Transform X to the equivalent uncorrelated standard normal space Y. Suppose all the variables X are uncorrelated, and we have

$$Y_i = \frac{X_i - \mu_{X_i}^N}{\sigma_{X_i}^N} \tag{3.12}$$

where $\mu_{X_i}^N$ and $\sigma_{X_i}^N$ are respectively the equivalent normal mean and standard deviation of X_i at the design point x_i^*, X_i is the random variable in the original space, and Y_i refers to the random variable

in the equivalent uncorrelated standard normal space. The transformation from X_i to Y_i for correlated variables was discussed in Shinozuka (1983) and Haldar and Mahadevan (2000).

Step 2. Transform Y space to Y' space using the following orthogonal transformation:

$$Y' = RY \tag{3.13}$$

where R is the rotation matrix. For the case of two random variables, it can be expressed as follows.

$$R = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \tag{3.14}$$

where θ is the angle of rotation. For the case of more than two random variables, the R matrix is computed in two steps (Haldar and Mahadevan, 2000).

Step 3. Compute matrix A whose elements are computed as follows.

$$a_{ij} = \frac{(RDR^t)_{ij}}{|\nabla G(\mathbf{y}^*)|} \quad i,j = 1,2,\ldots,n-1 \tag{3.15}$$

where D is the $n \times n$ second-order derivative matrix of the limit-state surface in the standard normal space evaluated at the design point; R is the rotation matrix; and $|\nabla G(\mathbf{y}^*)|$ is the length of the gradient vector in the standard normal space. D and $|\nabla G(\mathbf{y}^*)|$ are calculated using RBF network.

Step 4. Compute the eigenvalues of the matrix A for the principal curvature k_i. Once the k_i's and β are computed, Eq. (3.10) can be used to compute the second-order estimate of the probability of failure.

3.4. RBF based reliability analysis with correlated variables

Consider the X_i's in Eq. (3.2) to be correlated variables with means μ_{X_i}, standard deviation σ_{X_i}, and the covariance matrix represented as

$$[C] = \begin{bmatrix} \sigma_{X_1}^2 & \mathrm{Cov}(X_1,X_2) & \ldots & \mathrm{Cov}(X_1,X_n) \\ \mathrm{Cov}(X_2,X_1) & \sigma_{X_2}^2 & \ldots & \mathrm{Cov}(X_2,X_n) \\ \vdots & \vdots & \vdots & \vdots \\ \mathrm{Cov}(X_n,X_1) & \mathrm{Cov}(X_n,X_1) & \ldots & \sigma_{X_n}^2 \end{bmatrix} \tag{3.16}$$

The reduced variables X_i' are defined as

$$X_i' = \frac{X_i - \mu_{X_i}}{\sigma_{X_i}} \quad i = 1,2,\ldots,n \tag{3.17}$$

Then the covariance matrix of the reduced variables X_i' is

$$[C'] = \begin{bmatrix} 1 & \rho_{X_1,X_2} & \cdots & \rho_{X_1,X_n} \\ \rho_{X_2,X_1} & 1 & \cdots & \rho_{X_2,X_n} \\ \vdots & \vdots & \vdots & \vdots \\ \rho_{X_n,X_1} & \rho_{X_n,X_1} & \cdots & 1 \end{bmatrix} \tag{3.18}$$

where ρ_{X_i,X_j} is the correlation coefficient of the X_i and X_j variables.

The RBF based FORM and SORM methods can be used if the X_i's are transformed into uncorrelated reduced normal Z variables and Eq. (3.2) is expressed in terms of the Z variables. This can be done using the following equation (Haldar and Mahadevan, 2000):

$$\{X\} = \begin{bmatrix} & & 0 \\ & \sigma_X^N & \\ 0 & & \end{bmatrix} \cdot [T] \cdot [Z] + \{\sigma_X^N\} \tag{3.19}$$

where $\mu_{X_i}^N$ and $\sigma_{X_i}^N$ are respectively the equivalent normal mean and standard deviation of X_i at the design point, and T is a transformation matrix to convert the correlated reduced X' variables to uncorrelated reduced normal Z variables. The matrix containing the equivalent normal standard deviations in Eq. (3.19) is a diagonal matrix. The T matrix can be shown as

$$[T] = \begin{bmatrix} \theta_1^{(1)} & \theta_1^{(2)} & \dots & \theta_1^{(n)} \\ \theta_2^{(1)} & \theta_2^{(2)} & \dots & \theta_2^{(n)} \\ \vdots & \vdots & \vdots & \vdots \\ \theta_n^{(1)} & \theta_n^{(2)} & \dots & \theta_n^{(n)} \end{bmatrix} \tag{3.20}$$

$[T]$ is basically an orthogonal transformation matrix consisting of the eigenvectors of the correlation matrix $[C']$. $\{\theta^{(i)}\}$ is the eigenvector of the ith mode. $\theta_1^{(i)}, \theta_2^{(i)}, \dots, \theta_n^{(i)}$ are the components of the ith eigenvector.

Using Eq. (3.19), we can write Eq. (3.2) in terms of reduced uncorrelated normal Z variables. Then the RBF based FORM and SORM methods can be used in reliability analysis as described in Sections 3.2 and 3.3.

4. Applications

4.1. Example 1: A hypothetical nonlinear limit state

This example comes from Kaymaz (2005) and Kim and Na (1997) in which the limit state is defined as

$$g(X) = \exp[0.4(x_1 + 2) + 6.2] - \exp[0.3x_2 + 5.0] - 200 \tag{4.1}$$

where x_1 and x_2 are assumed to be independent and have a standard normal distribution with zero mean and unit standard deviation. The function $g(X)$ and its partial derivatives are approximated by a RBF network. The training set consists of 289 (17×17) points, which are uniformly spaced along x_1 and x_2 on $[-4,4]$. The test set contains 196 (14×14) points, which are also uniformly spaced along x_1 and x_2 on $[-4,4]$.

Parameters to be decided before the start of network training are the number of centers m, their locations $\{c^{(i)}\}_{i=1}^m$ and a set of the corresponding width $\{a^{(i)}\}_{i=1}^m$. According to Cover's Theorem (Haykin, 1999), the more basis functions are used, the better the approximation will be and so all data points will be taken to be the centers of the network ($m = n$) in this study. Thus $\{\mathbf{c}^{(i)} = \mathbf{x}^{(i)}\}_{i=1}^n$. The width of the ith basis function is determined according to the following relation

$$a^{(i)} = \eta d^{(i)} \tag{4.2}$$

where η is a factor, $\eta > 0$, and $d^{(i)}$ is the distance from the ith center to the nearest neighboring center. As a measure of the accuracy of different approximate schemes, a standard error norm of the solution, N_e, is defined as

$$N_e = \frac{\sqrt{\sum_{i=1}^{n_t} \left(y^{(i)} - f^{(i)}\right)^2}}{n_t} \tag{4.3}$$

where $f^{(i)}$ and $y^{(i)}$ are the calculated and exact function values at the point i, and n_t is the total number of the testing nodes. Smaller N_e s indicates more accurate approximations.

Table 3
N_e of the approximate function and its derivatives for $\eta = 8(5, 2)$ with RBF method 1

Original function	Gaussians	Multiquadrics	Inverse multiquadrics
	0.0952 (0.0678, 0.0503)	0.1446 (0.0631, 0.0564)	0.1033 (0.0465, 0.2297)
First derivative			
$\frac{\partial f}{\partial x_1}$	0.3204 (0.3025, 0.1800)	0.5295 (0.2183, 0.0964)	0.3943 (0.1553, 0.3874)
$\frac{\partial f}{\partial x_2}$	0.0752 (0.0385, 0.1011)	0.1157 (0.0631, 0.1503)	0.0852 (0.0348, 0.2317)
Second derivative			
$\frac{\partial^2 f}{\partial x_1^2}$	1.0558 (1.5174, 2.1357)	2.1457 (1.5397, 2.0887)	1.9993 (1.5181, 8.7112)
$\frac{\partial^2 f}{\partial x_2^2}$	0.0836 (0.1020, 0.9752)	0.3082 (0.3408, 1.3151)	0.2697 (0.1922, 2.2098)
$\frac{\partial^2 f}{\partial x_1 \partial x_2}$	0.2060 (0.1438, 0.1565)	0.1776 (0.1238, 0.1658)	0.2166 (0.1636, 0.4853)

Note: The numbers in parenthesis correspond to $\eta = 5$ and $\eta = 2$, respectively.

Table 3 shows the standard error norms N_e's of the approximate function and its first and second derivatives that are obtained from the RBF network method 1 at different factors ($\eta = 8, 5, 2$), using different types of radial basis function based on the 289 testing points. It can be seen that the errors of the approximate original function are the lowest, then the approximate first derivatives, and the errors of the approximate second derivatives are the highest. Fig. 3 clearly illustrates this very good approximation phenomenon of the exact and the approximate function, its first and second derivatives, in which $\eta = 2$ and Gaussian radial basis function is used.

Reliability analysis using RBF based FORM of this problem can be performed. The computation steps using Multiquadrics (MQ) in RBF method 1 are listed in Table 4. The computation steps using Inverse Multiquadrics (IMQ) and Gaussian (Gau) in RBF method 1 are omitted for simplicity, but the results are listed in Table 6. The reliability index $\beta = 2.7099$, and probability of failure $p_f = 0.003365$. Results from FORM method using the analytical first order derivatives are listed in Table 5, in which $\beta = 2.7099$ and $p_f = 0.003365$. The computations of RBF based FORM and analytical FORM both have 6 cycles. The probability of failure and reliability index of the same problem can also be computed using RBF based MCS with 100000 simulations. RBF network method 1 is used in RBF network. The same problem is also solved by Adaptive Monte-Carlo simulation, the classical response surface method, and the kriging method (Kaymaz, 2005; Kim and Na, 1997). All the results are summarized in Table 6. Probability of failure, reliability index and design points are compared. Conclusions could be drawn from these results that the proposed methods result in quite comparable accuracy.

4.2. Example 2: For RBF based MCS

The example to illustrate the RBF based MCS is the reliability analysis of the frame structure in Fig. 4. The performance function for this structural safety may be defined as.

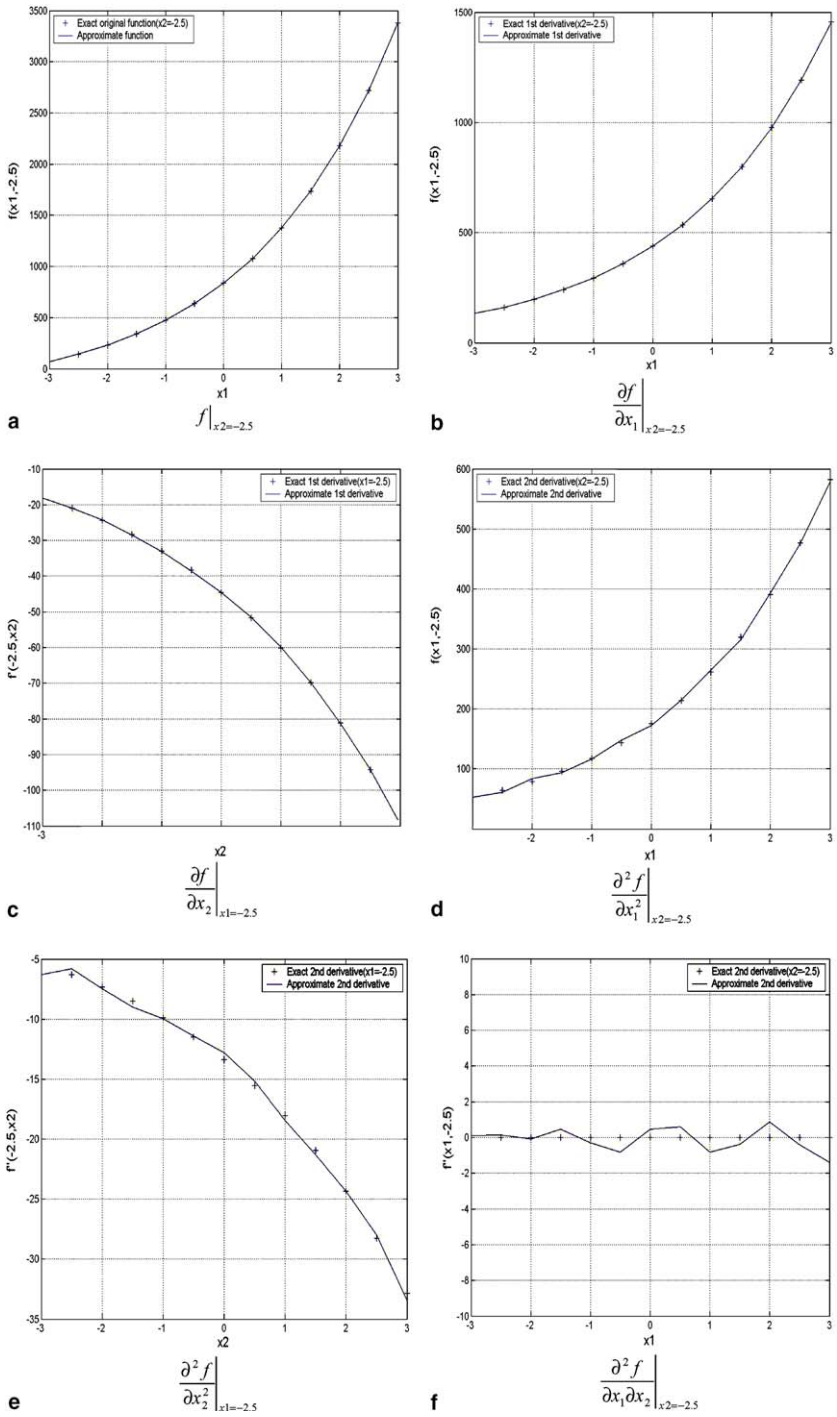

Fig. 3. The exact and the approximate function, its first and second derivatives, in which $\eta = 2$ and Gaussian radial basis function is used. The original function is $g(X) = \exp[0.4(x_1 + 2) + 6.2] - \exp[0.3x_2 + 5.0] - 200$.

Table 4
RBF method 1 based FORM using multiquadrics for Example 1

Step 1	$g(X) = \exp[0.4(x_1 + 2) + 6.2] - \exp[0.3x_2 + 5.0] - 200$						
Step 2	Initial values: $x_1^* = 0$, $x_2^* = 0$, $g(\ ) = 748.2200$						
Step 3	$\mu_{x_1}^N$	0	0	0	0	0	0
	$\sigma_{x_1}^N$	1	1	1	1	1	1
	$\mu_{x_2}^N$	0	0	0	0	0	0
	$\sigma_{x_2}^N$	1	1	1	1	1	1
	$x_1'^*$	0	−1.6938	−2.5208	−2.5821	−2.5423	−2.5383
	$x_2'^*$	0	0.1722	0.5296	0.8427	0.9382	0.9490
Step 4	$\left(\frac{\partial g}{\partial x_1}\right)^*$	437.2296	222.4450	159.7371	155.4137	157.9674	158.2267
	$\left(\frac{\partial g}{\partial x_2}\right)^*$	−44.4441	−46.7345	−52.1343	−57.3564	−59.0628	−59.2571
Step 5	$\left(\frac{\partial g}{\partial x_1'}\right)^*$	437.2296	222.4450	159.7371	155.4137	157.9674	158.2267
	$\left(\frac{\partial g}{\partial x_2'}\right)^*$	−44.4441	−46.7345	−52.1343	−57.3564	−59.0628	−59.2571
Step 6	New $x_1'^*$	−1.6938	−2.5208	−2.5821	−2.5423	−2.5383	−2.5378
	New $x_2'^*$	0.1722	0.5296	0.8427	0.9382	0.9490	0.9504
Step 7	New β	1.7025	2.5759	2.7161	2.7099	2.7099	2.7099
	$\Delta\beta$		0.8734	0.1403	0.0062	9.1378e−6	8.1353e−6
Step 8	New x_1^*	−1.6938	−2.5208	−2.5821	−2.5423	−2.5383	−2.5378
	New x_2^*	0.1722	0.5296	0.8427	0.9382	0.9490	0.9504
	New $g(\ )$	200.6780	26.1122	−0.7089	0.0026	0.0030	4.2863e−4
Computational cycle		I	II	III	IV	V	VI

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (−2.5378, 0.9504), $\beta = 2.7099$, $p_f = 0.003365$, Multiquadrics was used in RBF method 1. $a^{(i)} = 3$.

$$g(X) = 0.096 - u_A(A_1, A_2, A_3, A_4, A_5, P) \tag{4.4}$$

where $u_3(X)$ denotes the horizontal displacement (unit: m) at the node A as the function of basic random variables. In this equation, $g(X) < 0$ indicates failure.

The six basic random variables include the column and bean cross-section areas A_1, A_2, A_3, A_4, A_5 and the wind load P. The statistical parameters of the basic random variables are listed in Table 7. All the variables are assumed to be uncorrelated. The Young's modulus of all the members is assumed to be deterministic and is equal to 2.0×10^7 kN/m^2. The moments of inertia of the beam and the columns correlates with the cross-section areas as follows

Table 5
FORM method 2 for Example 1 using analytical derivatives

Step 1	$g(X) = \exp[0.4(x_1 + 2) + 6.2] - \exp[0.3x_2 + 5.0] - 200$					
Step 2	Initial values: $x_1^* = 0$, $x_2^* = 0$, $g(\ ) = 748.2200$					
Step 3	$\mu_{x_1}^N$	0	0	0	0	0
	$\sigma_{x_1}^N$	1	1	1	1	1
	$\mu_{x_2}^N$	0	0	0	0	0
	$\sigma_{x_2}^N$	1	1	1	1	1
	$x_1'^*$	0	−1.6883	−2.5178	−2.5824	−2.5442
	$x_2'^*$	0	0.1714	0.5286	0.8409	0.9337
Step 4	$\left(\frac{\partial g}{\partial x_1}\right)^*$	438.6533	223.2693	160.2291	156.1379	158.5444
	$\left(\frac{\partial g}{\partial x_2}\right)^*$	−44.5239	−46.8728	−52.1748	−57.2999	−58.9170
Step 5	$\left(\frac{\partial g}{\partial x_1'}\right)^*$	438.6533	223.2693	160.2291	156.1379	158.5444
	$\left(\frac{\partial g}{\partial x_2'}\right)^*$	−44.5239	−46.8728	−52.1748	−57.2999	−58.9170
Step 6	New $x_1'^*$	−1.6883	−2.5178	−2.5824	−2.5442	−2.5402
	New $x_2'^*$	0.1714	0.5286	0.8409	0.9337	0.9440
Step 7	New β	1.6970	2.5726	2.7159	2.7101	2.7099
	$\Delta\beta$		0.8756	0.1432	0.0058	1.9253e−4
Step 8	New x_1^*	−1.6883	−2.5178	−2.5824	−2.5442	−2.5402
	New x_2^*	0.1714	0.5286	0.8409	0.9337	0.9440
	New $g(\ )$	201.9305	26.6568	−0.6550	−0.0288	−4.2763e−4
Computational cycle		I	II	III	IV	V

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (−2.5402, 0.9440), $\beta = 2.7099$, $p_f = 0.003365$.

$$I_i = \alpha_i A_i^2 \quad (i = 1, 2, \ldots, 5) \tag{4.5}$$

where I_i are the moments of inertia and α_i are coefficients whose values are listed in Table 7.

The performance function of this problem does not explicitly contain any of the six basic random variables. The response variable u_A is dependent on the random variables and the deterministic variables, which cannot be expressed as a closed-form function. Instead, it has to be evaluated using the FEM. The performance function is implicit. A three-layer RBF network is established to represent the implicit performance function. The input layer has 6 neurons, and the output layer 1 neuron. The six variables in Table 7 are designed as the input variables. The performance function is the output variable. The training and testing samples consist of 257 patterns and 20 patterns. The RBF training is to determine the RBF unknown weights, and singular value decomposition method is used.

One hundred thousand sample values of the six basic random variables were generated according to their respective probabilistic distributions. These values were fed into the established RBF as input vectors. Then the RBF outputs 100,000 values of the performance function corresponding to the input vectors

Table 6
Results of Example 1 using different methods

Methods		Probability of failure	Reliability index	Design point
Adaptive Monte-Carlo simulation		0.00358	2.689	(−2.531, 0.9693)
Classical response surface method		0.003892	2.661	(−2.533, 0.815)
The kriging method		0.003051	2.742	(−2.648, 0.710)
FORM (analytical derivatives)		0.0033651	2.7099	(−2.5402, 0.944)
RBF based FORM (RBF network method 1)	IMQ	0.0033649	2.7099	(−2.5434, 0.9354)
	MQ	0.0033650	2.7099	(−2.5378, 0.9504)
	Gau	0.0033651	2.7099	(−2.5373, 0.9515)
RBF based MCS (RBF network method 1)	IMQ	0.003603	2.6872	
	MQ	0.003839	2.6659	
	Gau	0.003981	2.6537	

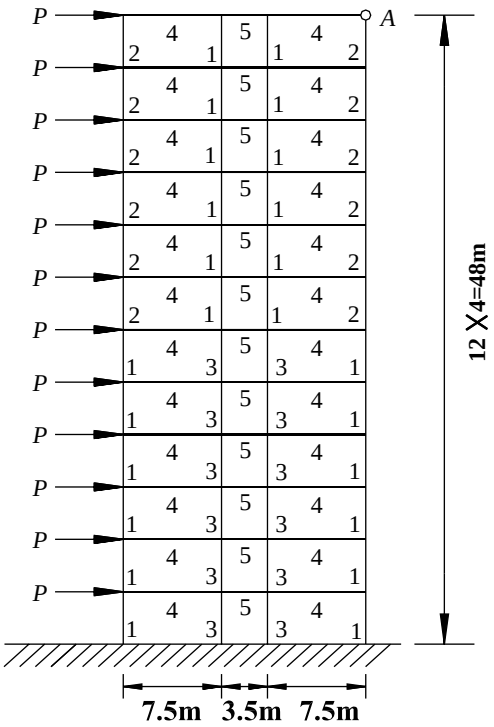

Fig. 4. Frame structure for Example 2.

accordingly. From these 100,000 values, the probabilistic characteristics (e.g., CDF or PDF) of the performance function can be extracted. And the probability of failure is estimated to be

$$p_f = P[g(X) < 0] = 0.07405 \tag{4.6}$$

The result of the same problem from the modified response surface method without using mixed terms is $p_f = 0.07309$ (Zhao, 1996). The result from Monte-Carlo simulation based on importance sampling using 2000 simulations is $p_f = 0.07506$ (Zhao, 1996). It is evident that these results correspond quite well. The proposed RBF based MCS combines the advantages of the conventional MCS and the RBF technique.

Table 7
Statistical data for Example 2 of RBF based MCS

Variables	Mean	Standard deviation	Type of distribution	Coefficient α_i
A_1	0.25 m^2	0.025	Lognormal	0.08333
A_2	0.16 m^2	0.016	Lognormal	0.08333
A_3	0.36 m^2	0.036	Lognormal	0.08333
A_4	0.20 m^2	0.020	Lognormal	0.26670
A_5	0.15 m^2	0.015	Lognormal	0.20000
P	30 kN	7.50	Gumbel (Fisher–Tipett extreme value type 1)	

Consequently, the proposed methodology is applicable to structural reliability problems with a wide arrange of variations including the number of random variables, the random variable distributions and the performance function.

4.3. Example 3: For RBF based FORM

The second example has been examined by Haldar and Mahadevan (2000) using the FORM method 2. A W16 × 31 steel section made of A36 steel is suggested to carry an applied deterministic bending moment of 1140 kip-in. The nominal yield stress F_y of the steel is 38 ksi. The nominal plastic modulus of the section Z is 54 in^3. F_y is assumed to be a lognormal variable with a mean of 38 ksi and a standard deviation of 3.8 ksi. Z is a normal variable with a mean of 54 in^3 and a standard deviation of 2.7 in^3. Consider the strength limit state equation:

$$g(\cdot) = F_y Z - 1140 = 0 \tag{4.7}$$

The computation steps using the RBF based FORM are listed in Table 8 RBF network method 1 and RBF network method 2 are applied to compute the partial derivatives of performance function.

In RBF network method 1, a three-layer RBF network is established to represent the performance function, and the first order derivatives are computed by the RBF network using differential rule. The input layer has 2 neurons, and the output layer 1 neuron. F_y and Z are designed as the input variables. The performance function is the output variable. RBF network is trained on a small set of data with different input values, with a set of 625 training samples, uniformly spaced within $[\mu_1-6\sigma_1, \mu_1 + 6\sigma_1]$ and a distance of $\frac{\sigma_1}{2}$, and within $[\mu_2-6\sigma_2, \mu_2 + 6\sigma_2]$ and a distance of $\frac{\sigma_2}{2}$. Here μ_1 and μ_2 are the mean values of F_y and Z, respectively; σ_1 and σ_2 are the standard deviation values of F_y and Z, respectively.

In RBF network method 2, a three-layer RBF network is established to represent the first order partial derivative of $f(x_1,x_2)$ with respect to x_1, and the original function $f(x_1,x_2)$ is calculated by the integral rule. The input layer has 2 neurons, and the output layer 1 neuron. F_y and Z are designed as the input variables. The first order partial derivative is the output variable. RBF network is trained on a small set of data with different input values, with a set of 625 training samples, uniformly spaced within $[\mu_1-6\sigma_1, \mu_1 + 6\sigma_1]$ and a distance of $\frac{\sigma_1}{2}$, and within $[\mu_2-6\sigma_2, \mu_2 + 6\sigma_2]$ and a distance of $\frac{\sigma_2}{2}$. Here μ_1 and μ_2 are the mean values of F_y and Z, respectively; σ_1 and σ_2 are the standard deviation values of F_y and Z, respectively. Another three-layer RBF network is established to represent the first order partial derivative of $f(x_1,x_2)$ with respect to x_2, and the original function $f(x_1,x_2)$ is calculated by the integral rule.

In the model of RBF network method 1, inverse multiquadrics (IMQ), multiquadrics (MQ), and Gaussian (Gau) radial basis function were respectively used. In the model of RBF network method 2, only inverse multiquadrics and multiquadrics radial basis function were used because Gaussian radial basis function isn't able to be integrated analytically.

Tables 8(a), 8(b) and 8(c) showed the computation steps of the RBF based FORM, in which inverse multiquadrics, multiquadrics and Gaussian basis function was used respectively in RBF network method 1.

Table 8(a)
RBF method 1 based FORM using inverse multiquadrics for Example 3

Step 1	$g(\ ) = F_y Z - 1140$					
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$					
Step 3	$\mu_{F_y}^N$	37.8109	35.1146	34.9568	34.9998	35.0051
	$\sigma_{F_y}^N$	3.7906	2.4400	2.4045	2.4141	2.4153
	μ_Z^N	54.00	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70	2.70
	$f_y'^*$	0.0499	−4.3661	−4.5132	−4.4734	−4.4684
	z'^*	0	−1.7657	−2.4727	−2.5538	−2.5620
Step 4	$\left(\frac{\partial g}{\partial_y}\right)^*$	53.9830	49.3857	47.4422	47.2386	47.2181
	$\left(\frac{\partial g}{\partial Z}\right)^*$	37.9955	24.4583	24.1204	24.2163	24.2281
Step 5	$\left(\frac{\partial g}{\partial F_y'}\right)^*$	204.6255	120.5030	114.0743	114.0368	114.0437
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.5877	66.0375	65.1252	65.3840	65.4157
Step 6	New $f_y'^*$	−3.5218	−4.5121	−4.4733	−4.4684	−4.4679
	New z'^*	−1.7657	−2.4727	−2.5538	−2.5620	−2.5628
Step 7	New β	3.9397	5.1453	5.1510	5.1508	5.1508
	$\Delta\beta$		1.2056	0.0057	1.6048e−4	1.7109e−4
Step 8	New f_y^*	24.4612	24.1048	24.2008	24.2127	24.2139
	New z^*	49.2327	47.3237	47.1047	47.0826	47.0804
	New $g(\ )$	64.2919	0.7294	−0.0290	−0.0015	−1.2623e−4
Computational cycle		I	II	III	IV	V

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2139, 47.0804), $\beta = 5.1508$. Inverse multiquadrics was used in RBF method 1. $a^{(i)} = 6$.

These tables show that although different radial basis functions were used, the computation results were identical, i.e., the reliability index $\beta = 5.1508$. However, when multiquadrics basis function was used, the computation cycles were relatively small.

Tables 8(d) and 8(e) showed the computation steps of the RBF based FORM, in which inverse multiquadrics and multiquadrics basis function were used respectively in RBF network method 2. In both cases there are four computation cycles. In Table 8(a) and Table 8(d) inverse multiquadrics basis function was both used, the computation cycle in Table 8(d) is smaller than that of Table 8(a). The reason was probably due to the derivative computation methods. RBF network method 2 was used in Table 8(d) while RBF network method 1 was used in Table 8(a). RBF network method 2 gives more accurate derivatives of performance function, as also pointed out by Mai-Duy and Tran-Cong (2003).

The computation steps of the FORM method 2 using analytical derivatives are listed in Table 9 for comparison (Haldar and Mahadevan, 2000). The solution is obtained after four iterations and the convergence criteria are satisfied. The final checking point is (24.22, 47.07). The reliability index is $\beta = 5.151$. The results

Table 8(b)
RBF method 1 based FORM using multiquadrics for Example 3

Step 1	$g(\ ) = F_y Z - 1140$				
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$				
Step 3	$\mu_{F_y}^N$	37.8109	35.1148	34.9596	35.0022
	$\sigma_{F_y}^N$	3.7906	2.4401	2.4051	2.4146
	μ_Z^N	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70
	$f_y'^*$	0.0499	−4.3659	−4.5106	−4.4712
	z'^*	0	−1.7651	−2.4780	−2.5575
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	53.9910	49.2554	47.3139	47.1009
	$\left(\frac{\partial g}{\partial Z}\right)^*$	37.9905	24.4608	24.1080	24.2068
Step 5	$\left(\frac{\partial g}{\partial F'_y}\right)^*$	204.6558	120.1875	113.7954	113.7293
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.5743	66.0441	65.0916	65.3584
Step 6	New $f_y'^*$	−3.5217	−4.5096	−4.4711	−4.4659
	New z'^*	−1.7651	−2.4780	−2.5575	−2.5665
Step 7	New β	3.9393	5.1456	5.1509	5.1508
	$\Delta\beta$		1.2063	0.0053	1.0695e−4
Step 8	New f_y^*	24.4617	24.1111	24.2061	24.2189
	New z^*	49.2342	47.3093	47.0947	47.0706
	New $g(\ )$	64.3535	0.6804	−0.0215	−3.7307e−4
Computational cycle		I	II	III	IV

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2189, 47.0706), $\beta = 5.1508$. Multiquadrics was used in RBF method 1. $a^{(i)} = 6$.

for the same problem using multilayer perceptrons (Deng et al., 2005) are also listed in Table 10. The first-order partial derivatives of the performance function were computed using a multilayer perceptron artificial neural network. The final checking point is (24.2888, 46.9351). The reliability index is $\beta = 5.15105$.

All the results are summarized in Table 11 in which reliability index, design points and computational cycle are compared. It can be concluded that both FORM methods have similar results. The difference of the results mainly results from the different computation parameters and errors. Multilayer perceptrons (MLP) method from Deng et al. (2005) has six computation cycles, while RBF methods have 4–5 cycles, analytical derivatives method (i.e. FORM method 2 in Haldar and Mahadevan, 2000) has only four cycles. However, computer programs are used for these computations, and no extra work is performed manually.

4.4. Example 4: For RBF based SORM

Reconsider the above problem and the strength limit state equation is

$$g(\cdot) = F_y Z - 1140 = 0 \tag{4.8}$$

Table 8(c)
RBF method 1 based FORM using Gaussian for Example 3

Step 1	$g(\ ) = F_y Z - 1140$					
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$					
Step 3	$\mu_{F_y}^N$	37.8109	35.0954	34.9418	35.0033	35.0126
	$\sigma_{F_y}^N$	3.7906	2.4356	2.4012	2.4148	2.4169
	μ_Z^N	54.00	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70	2.70
	$f_y'^*$	0.0499	−4.3841	−4.5270	−4.4701	−4.4615
	z'^*	0	−1.7597	−2.4443	−2.5591	−2.5740
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	53.9502	50.1673	48.2465	48.0458	48.0162
	$\left(\frac{\partial g}{\partial Z}\right)^*$	37.7189	24.4409	24.5643	24.7924	24.8035
Step 5	$\left(\frac{\partial g}{\partial F_y'}\right)^*$	204.5009	122.1898	115.8483	116.0229	116.0518
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	101.8412	65.9904	66.3237	66.9396	66.9695
Step 6	New $f_y'^*$	−3.5335	−4.5260	−4.4700	−4.4615	−4.4613
	New z'^*	−1.7597	−2.4443	−2.5591	−2.5740	−2.5744
Step 7	New β	3.9474	5.1439	5.1507	5.1508	5.1508
	$\Delta\beta$		1.1965	0.0068	6.8544e−5	9.2002e−5
Step 8	New f_y^*	24.4172	24.0716	24.2086	24.2296	24.2301
	New z^*	49.2489	47.4003	47.0905	47.0501	47.0490
	New $g(\ )$	62.5209	1.0001	−0.0057	0.0027	1.3373e−4
Computational cycle		I	II	III	IV	V

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2301, 47.0490), $\beta = 5.1508$. Gaussian was used in RBF method 1. $a^{(i)} = 6$.

Assume that F_y is a lognormal variable with a mean of 38 ksi and a standard deviation of 3.8 ksi. Z is a normal variable with a mean of 54 in^3 and a standard deviation of 2.7 in^3. SORM method is used to calculate the reliability index. The computation steps of failure probability using RBF network method 3 is discussed in detail.

In Table 9, we find that the final checking point in FORM of the original variable space using RBF method 2 is (24.2192, 47.0700). The equivalent normal mean and standard deviation of F_y at the design point are 35.0025 and 2.4147, respectively. In the standard normal space, the design point is defined as:

$$y_{F_y}^* = \frac{24.2192 - 35.0025}{2.4147} = -4.4709 \tag{4.9a}$$

$$y_Z^* = \frac{47.0700 - 54}{2.7} = -2.5580 \tag{4.9b}$$

The direction cosines for F_y and Z at the original design point are $\frac{113.7150}{\sqrt{113.7150^2+65.3569^2}} = 0.8670$ and $\frac{65.3569}{\sqrt{113.7150^2+65.3569^2}} = 0.4983$, respectively. Consequently, the R matrix is defined as.

Table 8(d)
RBF method 2 based FORM using inverse multiquadrics for Example 3

Step 1	$g(\ ) = F_y Z - 1140$				
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$				
Step 3	$\mu_{F_y}^N$	37.8109	35.1160	34.9602	35.0025
	$\sigma_{F_y}^N$	3.7906	2.4404	2.4052	2.4147
	μ_Z^N	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70
	$f_y'^*$	0.0499	−4.3648	−4.5101	−4.4709
	z'^*	0	−1.7650	−2.4789	−2.5580
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	53.9988	49.2340	47.3074	47.0937
	$\left(\frac{\partial g}{\partial Z}\right)^*$	38.0016	24.4644	24.1119	24.2063
Step 5	$\left(\frac{\partial g}{\partial F'_y}\right)^*$	204.6851	120.1488	113.7854	113.7150
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.6044	66.0538	65.1021	65.3569
Step 6	New $f_y'^*$	−3.5210	−4.5091	−4.4708	−4.4657
	New z'^*	−1.7650	−2.4789	−2.5580	−2.5667
Step 7	New β	3.9386	5.1455	5.1509	5.1508
	$\Delta\beta$		1.2069	0.0053	9.8073e−5
Step 8	New f_y^*	24.4644	24.1123	24.2068	24.2192
	New z^*	49.2345	47.3069	47.0935	47.0700
	New $g(\ )$	64.4947	0.6785	−0.0203	−3.0669e−4
Computational cycle		I	II	III	IV

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2192, 47.0700), $\beta = 5.1508$. Inverse multiquadrics was used in RBF method 2. $a^{(i)} = 6$.

$$R = \begin{bmatrix} 0.4983 & -0.8670 \\ 0.8670 & 0.4983 \end{bmatrix}. \tag{4.10}$$

Using the chain rule of differentiation, the second order derivatives of the performance function in the equivalent standard normal space are

$$\frac{\partial^2 g(\ )}{\partial^2 F'_y} = \frac{\partial^2 g(\ )}{\partial^2 F_y} \cdot (\sigma_{F_y}^N)^2 \tag{4.11a}$$

$$\frac{\partial^2 g(\ )}{\partial^2 Z'} = \frac{\partial^2 g(\ )}{\partial^2 Z} \cdot (\sigma_Z^N)^2 \tag{4.11b}$$

$$\frac{\partial^2 g(\ )}{\partial F'_y \partial Z'} = \frac{\partial^2 g(\ )}{\partial F_y \partial Z} \cdot (\sigma_{F_y}^N \sigma_Z^N) \tag{4.11c}$$

Table 8(e)
RBF method 2 based FORM using multiquadrics for Example 3

Step 1	$g(\ ) = F_y Z - 1140$				
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$				
Step 3	$\mu_{F_y}^N$	37.8109	35.1161	34.9601	35.0025
	$\sigma_{F_y}^N$	3.7906	2.4404	2.4052	2.4147
	μ_Z^N	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70
	$f_y'^*$	0.0499	−4.3646	−4.5102	−4.4709
	z'^*	0	−1.7647	−2.4788	−2.5580
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	54.0025	49.2342	47.3102	47.0964
	$\left(\frac{\partial g}{\partial Z}\right)^*$	37.9994	24.4629	24.1136	24.2086
Step 5	$\left(\frac{\partial g}{\partial F_y'}\right)^*$	204.6993	120.1506	113.7910	113.7219
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.5985	66.0497	65.1067	65.3633
Step 6	New $f_y'^*$	−3.5209	−4.5091	−4.4708	−4.4657
	New z'^*	−1.7647	−2.4788	−2.5580	−2.5667
Step 7	New β	3.9384	5.1456	5.1509	5.1508
	$\Delta\beta$		1.2071	0.0053	9.7103e−5
Step 8	New f_y^*	24.4647	24.1121	24.2068	24.2193
	New z^*	49.2352	47.3073	47.0933	47.0698
	New $g(\ )$	64.5246	0.6760	−0.0202	−2.8941e−4
Computational cycle		I	II	III	IV

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2193, 47.0698), $\beta = 5.1508$. Multiquadrics was used in RBF method 2. $a^{(i)} = 6$.

The second order derivatives of the performance function in the original space can be obtained by the RBF network method 3 as follows.

$$\frac{\partial^2 g(\ )}{\partial^2 F_y} = 6.9067e - 6 \tag{4.12a}$$

$$\frac{\partial^2 g(\ )}{\partial^2 Z} = 0.0010 \tag{4.12b}$$

$$\frac{\partial^2 g(\ )}{\partial F_y \partial Z} = 1.0078. \tag{4.12c}$$

Therefore, the matrix D is found to be as follows.

$$D = \begin{bmatrix} 6.9067e-6 \times 2.4147^2 & 1.0078 \times 2.4147 \times 2.7 \\ 1.0078 \times 2.4147 \times 2.7 & 0.0010 \times 2.7^2 \end{bmatrix} = \begin{bmatrix} 0.0000 & 6.5705 \\ 6.5705 & 0.0073 \end{bmatrix} \tag{4.13}$$

Table 9
FORM method 2 for Example 3, after Haldar and Mahadevan (2000)

Step 1	$g(\ ) = F_y Z - 1140$				
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$				
Step 3	$\mu_{F_y}^N$	37.81	35.116	34.960	35.003
	$\sigma_{F_y}^N$	3.79	2.44	2.405	2.415
	μ_Z^N	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70
	$f_y'^*$	0.05	−4.365	−4.510	−4.471
	z'^*	0.00	−1.765	−2.479	−2.558
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	54.00	49.235	47.307	47.093
	$\left(\frac{\partial g}{\partial Z}\right)^*$	38.00	24.464	24.112	24.207
Step 5	$\left(\frac{\partial g}{\partial F_y'}\right)^*$	204.69	120.15	113.78	113.71
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.60	66.05	65.10	65.36
Step 6	New $f_y'^*$	−3.521	−4.509	−4.471	−4.466
	New z'^*	−1.765	−2.479	−2.558	−2.567
Step 7	New β	3.939	5.145	5.151	5.151
	$\Delta\beta$		1.206	0.006	0.0001
Step 8	New f_y^*	24.464	24.112	24.207	24.22
	New z^*	49.235	47.307	47.093	47.07
	New $g(\ )$	64.500	0.679	−0.020	−0.0002
Computational cycle		I	II	III	IV

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.22, 47.07), $\beta = 5.151$.

Similarly, the first order derivatives of the performance function in the equivalent standard normal space are defined as

$$\frac{\partial g(\ )}{\partial F_y'} = \frac{\partial g(\ )}{\partial F_y}\frac{\partial F_y}{\partial F_y'} = \frac{\partial g(\ )}{\partial F_y}\sigma_{F_y}^N \tag{4.14a}$$

$$\frac{\partial g(\ )}{\partial Z'} = \frac{\partial g(\ )}{\partial Z}\frac{\partial Z}{\partial Z'} = \frac{\partial g(\ )}{\partial Z}\sigma_Z \tag{4.14b}$$

The first order derivatives of the performance function in the original space can be obtained by the RBF method as follows.

$$\frac{\partial g(\ )}{\partial F_y} = 47.0937 \tag{4.15a}$$

$$\frac{\partial g(\ )}{\partial Z} = 24.2063 \tag{4.15b}$$

Table 10
FORM using multilayer perceptrons for Example 3, after Deng et al. (2005)

Step 1	$g(\ ) = F_y Z - 1140$						
Step 2	Initial values: $f_y^* = 38$, $z^* = 54$, $g(\ ) = 912.0$						
Step 3	$\mu_{F_y}^N$	37.81	35.1154	34.9835	35.0331	35.0383	35.0389
	$\sigma_{F_y}^N$	3.79	2.4402	2.4104	2.4215	2.4227	2.4228
	μ_Z^N	54.00	54.00	54.00	54.00	54.00	54.00
	σ_Z^N	2.70	2.70	2.70	2.70	2.70	2.70
	$f_y'^*$	0.04987	−4.3653	−4.4885	−4.4424	−4.4375	−4.4370
	z'^*	0	−1.76476	−2.5145	−2.6056	−2.6154	−2.6165
Step 4	$\left(\frac{\partial g}{\partial F_y}\right)^*$	54	48.6835	46.7742	46.5658	46.5432	46.5407
	$\left(\frac{\partial g}{\partial Z}\right)^*$	38	24.6531	24.4928	24.6146	24.6276	24.6290
Step 5	$\left(\frac{\partial g}{\partial F_y'}\right)^*$	204.683	118.7990	112.7460	112.7603	112.7606	112.7604
	$\left(\frac{\partial g}{\partial Z'}\right)^*$	102.579	66.5633	66.1305	66.4594	66.4946	66.4984
Step 6	New $f_y'^*$	−3.52136	−4.4877	−4.4423	−4.4375	−4.4370	−4.43696
	New z'^*	−1.76476	−2.5145	−2.6056	−2.6154	−2.6165	−2.6166
Step 7	New β	3.9388	5.1441	5.15008	5.15094	5.1510	5.15105
	$\Delta\beta$		1.2053	0.00598	−0.00086	0.00006	0.00005
Step 8	New f_y^*	24.4631	24.1644	24.2756	24.2847	24.2887	24.28885
	New z^*	49.2351	47.2109	49.9648	46.9383	46.9354	46.9351
	New $g(\ )$	64.4422	0.8244	0.102006	0.01338	0.001479	0.000164
Computational cycle		I	II	III	IV	V	VI

Note: Convergence criteria in steps 7 and 8: (1) $|\Delta\beta| \leqslant 0.001$, (2) $|g(\ )| \leqslant 0.001$. The final checking point is (24.2888, 46.9351), $\beta = 5.15105$.

Table 11
Results of Example 3 using different methods

Methods		Reliability index	Design point	Computational cycle
RBF based FORM (RBF network method 1)	IMQ	5.1508	(24.2139, 47.0804)	5
	MQ	5.1508	(24.2189, 47.0706)	4
	Gau	5.1508	(24.2301, 7.0490)	5
RBF based FORM (RBF network method 2)	IMQ	5.1508	(24.2192, 47.0700)	4
	MQ	5.1508	(24.2193, 7.0698)	4
FORM method 2		5.151	(24.22, 47.07)	4
FORM using MLP		5.15105	(24.2888, 46.9351)	6

Therefore, at the design point, the gradient vector in the standard normal space can be determined as follows.

$$\nabla G(y^*) = \begin{Bmatrix} \frac{\partial g(\ )}{\partial F'_y} \\ \frac{\partial g(\ )}{\partial Z'} \end{Bmatrix} = \begin{Bmatrix} 47.0937 \times 2.4147 \\ 24.2063 \times 2.7 \end{Bmatrix} = \begin{Bmatrix} 113.7150 \\ 65.3569 \end{Bmatrix} \tag{4.16}$$

The length of this vector is

$$|\nabla G(y^*)| = \sqrt{(113.7150)^2 + (65.3569)^2} = 131.1588 \tag{4.17}$$

The matrix A is computed as follows:

$$\begin{aligned}[A] &= \frac{(RDR^t)}{|\nabla G(y^*)|} = \frac{1}{131.1588} \begin{bmatrix} 0.4983 & -0.8670 \\ 0.8670 & 0.4983 \end{bmatrix} \begin{bmatrix} 0.0000 & 6.5705 \\ 6.5705 & 0.0073 \end{bmatrix} \begin{bmatrix} 0.4983 & -0.8670 \\ 0.8670 & 0.4983 \end{bmatrix} \\ &= \begin{bmatrix} -0.0432 & -0.0252 \\ -0.0252 & -0.0433 \end{bmatrix} \end{aligned} \tag{4.18}$$

Furthermore, the principal curvature $k_1 = a_{11} = -0.041735$. Finally, the probability of failure using SORM in combination with RBF method is as follows.

$$p_{f_2} \approx \Phi(-5.1508)[1 + 5.1508 \times (-0.0432)]^{-\frac{1}{2}} = 1.4711 \times 10^{-7} \tag{4.19}$$

The safety index is determined as follows.

$$\beta_{\mathrm{SORM}} = -\Phi^{-1}(1.4711 \times 10^{-7}) = 5.1271. \tag{4.20}$$

The results of this example by Haldar and Mahadevan (2000) are as follows.

$$p_{f_2} \approx 1.4708 \times 10^{-7} \tag{4.21}$$

and

$$\beta_{\mathrm{SORM}} = 5.1272. \tag{4.22}$$

No significant differences can be observed between results in Eqs. (4.19) and (4.21) or between Eqs. (4.20) and (4.22).

If the partial derivatives were computed by RBF network method 1, the final checking point in FORM of the original variable space using RBF method 1 is (24.2139, 47.0804). The results of main computation steps are listed as follows.

$$R = \begin{bmatrix} 0.4976 & -0.8674 \\ 0.8674 & 0.4976 \end{bmatrix}$$

$$D = \begin{bmatrix} 0.1389 \times 2.4153^2 & 1.0029 \times 2.4153 \times 2.7 \\ 1.0029 \times 2.4153 \times 2.7 & 0.0118 \times 2.7^2 \end{bmatrix} = \begin{bmatrix} 0.8102 & 6.5404 \\ 6.5404 & 0.0858 \end{bmatrix}$$

$$\nabla G(y^*) = \begin{Bmatrix} 114.0437 \\ 65.4157 \end{Bmatrix}$$

$$[A] = \frac{(RDR^t)}{|\nabla G(y^*)|} = \begin{bmatrix} -0.0409 & -0.0227 \\ -0.0227 & 0.0477 \end{bmatrix}$$

$$p_{f_2} \approx \Phi(-5.1508)[1 + 5.1508 \times (-0.0409)]^{-\frac{1}{2}} = 1.4599 \times 10^{-7} \tag{4.23}$$

$$\beta_{\mathrm{SORM}} = 5.1285. \tag{4.24}$$

Table 12
Results of FORM and SORM for Example 3 and Example 4

Probability distributions		Reliability index					
F_y	Z	FORM			SORM		
		RBF based FORM[a]	MLP based FORM[b]	FORM Method 2[c]	RBF based SORM[a]	MLP based SORM[b]	SORM[c]
Normal	Normal	4.2409	4.2614	4.261	4.2456	4.2471	4.246
Lognormal	Normal	5.1508	5.1511	5.151	5.1285	5.1283	5.139
Normal	Lognormal	4.2659	4.2660	4.266	4.2560	4.2543	4.259
Lognormal	Lognormal	5.2133	5.2126	5.213	5.2105	5.2011	5.211

[a] RBF method 2 with Multiquadrics was used in RBF based FORM.
[b] MLP refers to multilayer perceptron artificial neural networks. The results come from Deng et al. (2005).
[c] The results come from Haldar and Mahadevan (2000).

Comparing Eqs. (4.20), (4.22) and (4.24), it can be concluded that RBF network method 1 based SORM and RBF network method 3 based SORM are both right and accurate enough. The differences between them come from the different parameters of the RBF networks.

4.5. Discussions

To illustrate the applicability of the RBF based FORM and SORM, different distributions of F_y and Z are considered in the calculation of the safety indices of the same beam problem as described in the above two sub-sections. Multiquadrics is used in RBF method 2. The results of RBF method 2 based FORM and SORM are listed in Table 12. For comparison, the results of Deng et al. (2005) and Haldar and Mahadevan (2000) are also listed. Again, no significant differences are observed. It is shown that the RBF based FORM or SORM can be used to calculate the probability of failure and the safety index. In our examples, simple problems with only two random variables are presented in the explicit performance functions. The performance functions are barely nonlinear. The simple examples are used for illustration and for comparison.

In the proposed RBF based approaches, unlike the response surface method, it is not necessary to know the underlying relationship or to suppose a relationship between the input variables and the output. The RBF is a universal approximator and can be used to approximate linear or non-linear, implicit or explicit performance functions. Therefore, the RBF based MCS, FORM or SORM can cope with problems whose performance functions are linear or non-linear, implicit or explicit in terms of multi-variable. The RBF based MCS, FORM or SORM are especially useful for reliability problems with implicit and nonlinear performance functions where other reliability methods are not applicable. Minor extra manual work is needed with the increase of random variable number since the key task is to establish an RBF model.

The computation CPU time and evaluation numbers of limit state (or performance) function are studied in detail in Gomes and Awruch (2004) about RBF network's approximation capability, and the following conclusions have been made: "*For examples with large structural systems with implicit limit state function (LSF) and high computational cost, techniques such as Monte-Carlo Simulation may be a feasible alternative if the LSF may be approximated using ANN*" (RBF). Emphases in this paper were placed on the proposed methodology. RBF networks are used to approximate and replace the performance functions or their partial derivatives. Thus the number of direct calls to the performance function was reduced. In the evaluation of RBF network, only simple mathematics and little CPU time are needed to obtain the outcome. Computation of derivatives is intrinsically to extract information from the established and successfully trained RBF network, so no extra evaluation numbers of performance function and computation CPU time are needed. Consequently, the conclusions of Gomes and Awruch (2004) and Hussain et al. (2002) on RBF networks are also feasible in this work.

5. Conclusions

The main contribution of this paper is to propose three RBF network methods to compute the (implicit and often complex) performance function derivatives and then to combine them with conventional MCS, FORM and SORM and propose three RBF reliability analysis methods: RBF based MCS, RBF based FORM and RBF based SORM. The presented methodology is also convenient for problems with explicit but highly non-linear performance functions or for problems with a large number of basic random numbers. Such problems may be difficult to solve by conventional approximation methods and simulations because of either prohibitive computational cost or loss of accuracy. RBF network is applied in these methods to simultaneously estimate the implicit performance function and its first or second order partial derivatives. Reliability analysis is performed on the RBF network instead of the real performance function. The approach is conceptually elegant. Illustrative examples, although simple, do show that the RBF based MCS, FORM, or SORM are feasible for reliability analysis. The focus of this paper were placed on the proposed RBF based methodology and algorithms. For more details on other RBF network issues, such as training sets selection, hidden neurons numbers, the training algorithm, and potential limitations of RBF, etc. refer to references (Haykin, 1999; Li, 1996; Warnes et al., 1998).

The RBF based MCS differentiates itself from other MCS methods in that it employs the robust generality capability of RBF to compute the values of implicit performance function, which combines the advantages of conventional MCS and RBF technique. It can thus prohibitively reduce the computation time. This approach is applicable to structural reliability problems with a wide arrange of variations including the number of random variables, the random variable distributions and the performance function. The RBF based FORM differentiates itself from other FORMs in that it employs an RBF to compute the values and gradients of implicit performance function. The RBF based SORM differentiates itself from other SORMs since it employs an RBF to compute the values, the first and second order partial derivatives of implicit performance function. The RBF based FORM or SORM are especially useful for reliability problems with implicit and nonlinear performance functions.

Acknowledgements

Financial supports from National Natural Science Foundation of China (No. 50401010 and No. 50490274) and China Postdoctoral Science Foundation were gratefully acknowledged by the author. Special thanks are due to Dr. Zhiwei Li from The Hong Kong Polytechnic University for his contributions to the computation in Appendix of this work. The author also expressed his appreciation to the anonymous reviewers and the Editors for their valuable comments and suggestions that help the author refine and greatly improve the quality of this paper.

Appendix A

In Section 2.2.3, the integral calculus $\int (x^2+a)\ln(b+\sqrt{x^2+c})\,\mathrm{d}x$ is calculated as

$$\begin{aligned}\int (x^2+a)\ln\left(b+\sqrt{x^2+c}\right)\mathrm{d}x &= \frac{1}{3}\int \ln\left(b+\sqrt{x^2+c}\right)\mathrm{d}x^3 + a\int \ln\left(b+\sqrt{x^2+c}\right)\mathrm{d}x \\ &= \frac{1}{3}\ln\left(b+\sqrt{x^2+c}\right)x^3 + a\ln\left(b+\sqrt{x^2+c}\right)x \\ &\quad -\frac{1}{3}\int x^3\,\mathrm{d}\left(\ln(b+\sqrt{x^2+c})\right) - a\int x\,\mathrm{d}\left(\ln(b+\sqrt{x^2+c})\right) \end{aligned} \tag{A.1}$$

Suppose

$$\mathrm{I1} = \int x^3 \,\mathrm{d}\left(\ln(b + \sqrt{x^2 + c})\right) \tag{A.2}$$

$$\mathrm{I2} = \int x \,\mathrm{d}\left(\ln(b + \sqrt{x^2 + c})\right) \tag{A.3}$$

In order to calculate the integral calculus (A.2), let

$$x = \sqrt{c}\,\mathrm{sh}\,t \tag{A.4}$$

where $\mathrm{sh}\,t = \frac{\mathrm{e}^t - \mathrm{e}^{-t}}{2}$, $\mathrm{ch}\,t = \frac{\mathrm{e}^t + \mathrm{e}^{-t}}{2}$, and $\sqrt{x^2 + c} = \sqrt{c}\mathrm{ch}\,t$.

(1) Calculation of I1

$$\begin{aligned}\mathrm{I1} &= \int x^3 \,\mathrm{d}\left(\ln(b + \sqrt{x^2 + c})\right) = \int \frac{(\sqrt{c}\,\mathrm{sh}\,t)^3}{b + \sqrt{c}\,\mathrm{ch}\,t}\,\mathrm{d}\sqrt{c}\,\mathrm{ch}\,t = (\sqrt{c})^3 \int \frac{(\mathrm{sh}\,t)^4}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t}\,\mathrm{d}t \\ &= (\sqrt{c})^3 \int \frac{[(\mathrm{ch}\,t)^2 - 1]^2}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t}\,\mathrm{d}t = (\sqrt{c})^3 \int \frac{(\mathrm{ch}\,t)^4 - 2(\mathrm{ch}\,t)^2 + 1}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t}\,\mathrm{d}t\end{aligned} \tag{A.5}$$

This expression can be expressed as

$$\frac{(\mathrm{ch}\,t)^4 - 2(\mathrm{ch}\,t)^2 + 1}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t} = A(\mathrm{ch}\,t)^3 + B(\mathrm{ch}\,t)^2 + F(\mathrm{ch}\,t) + D + \frac{E}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t} \tag{A.6}$$

where

$$A = 1, \quad B = -\frac{b}{\sqrt{c}}, \quad F = \frac{b^2}{c} - 2, \quad D = \frac{b}{\sqrt{c}}\left(2 - \frac{b^2}{c}\right), \quad E = 1 + \frac{b^2}{c}\left(\frac{b^2}{c} - 2\right) \tag{A.7}$$

$$\int (\mathrm{ch}\,t)^3\,\mathrm{d}t = \int [(\mathrm{sh}\,t)^2 + 1]\,\mathrm{d}\,\mathrm{sh}\,t = \frac{1}{3}(\mathrm{sh}\,t)^3 + \mathrm{sh}\,t \tag{A.8}$$

$$\int (\mathrm{ch}\,t)^2\,\mathrm{d}t = \frac{1}{8}(\mathrm{e}^{2t} - \mathrm{e}^{-2t} + 4) \tag{A.9}$$

$$\int (\mathrm{ch}\,t)\,\mathrm{d}t = \mathrm{sh}\,t \tag{A.10}$$

$$\int \frac{1}{\frac{b}{\sqrt{c}} + \mathrm{ch}\,t}\,\mathrm{d}t = \frac{2\sqrt{c}}{\sqrt{c - b^2}}\mathrm{atan}\left[\frac{b + \sqrt{c}\exp(t)}{\sqrt{c - b^2}}\right] \tag{A.11}$$

Then the integral I1 of Eq. (A.5) can be obtained by placing Eq. (A.7) into Eq. (A.6) and by using Eqs. (A.8)–(A.11). Note

$$t = \mathrm{arsh}\left(\frac{x}{\sqrt{c}}\right) = \ln\left(\frac{x}{\sqrt{c}} + \sqrt{\frac{x^2}{c} + 1}\right) \tag{A.12}$$

The integral I1 can be expressed in terms of variable x.

(2) Calculation of I2

Similarly, the integral I2 can be obtained as follows:

$$\mathrm{I2} = \int x\,\mathrm{d}(\ln(b+\sqrt{x^2+c})) = \int \frac{\sqrt{c}\,\mathrm{sh}\,t}{b+\sqrt{c}\mathrm{ch}\,t} d\sqrt{c}\mathrm{ch}\,t = (\sqrt{c})\int \frac{(\mathrm{sh}\,t)^4}{\frac{b}{\sqrt{c}}+\mathrm{ch}\,t}\,\mathrm{d}t$$
$$= (\sqrt{c})\int \frac{[(\mathrm{ch}\,t)^2-1]^2}{\frac{b}{\sqrt{c}}+\mathrm{ch}\,t}\,\mathrm{d}t = (\sqrt{c})\int \frac{(\mathrm{ch}\,t)^4-2(\mathrm{ch}\,t)^2+1}{\frac{b}{\sqrt{c}}+\mathrm{ch}\,t}\,\mathrm{d}t \tag{A.13}$$

Then the integral I2 of Eq. (A.13) can be obtained by placing Eq. (A.7) into Eq. (A.6) and by using Eqs. (A.8)–(A.11). Note Eq. (A.12), I2 can be expressed in terms of x.

After calculation I1 and I2, the integral calculus $\int(x^2+a)\ln(b+\sqrt{x^2+c})\,\mathrm{d}x$ can be obtained by using Eq. (A.1).

References

Ang, A.H.S., Tang, W.H., 1975. Probability Concepts in Engineering Planning and Design, Basic Principles, vol. 1. John Wiley & Sons, Inc., NewYork.

Ang, A.H.S., Tang, W.H., 1984. Probability Concepts in Engineering Planning and Design, Decision, Risk, and Reliability, vol. 2. John Wiley & Sons, Inc., NewYork.

Anjum, M.F., Imran, T., Khaled, A.-S., 1997. Response surface methodology: a neural network approach. European Journal of Operational Research. 101, 65–73.

Bauer, J., Pula, W., 2000. Reliability with respect to settlement limit states of shallow foundations on linearly deformable subsoil. Computers and Geotechnics 26 (3–4), 281–308.

Bishop, C.M., 1995. Neural Networks for Pattern Recognition. Oxford University Press.

Breitung, K., 1984. Asymptotic approximation for multinormal integrals. Journal of Engineering Mechanics Division, ASCE 110 (3), 357–366.

Bucher, C.G., Bourgund, U., 1990. A fast and efficient response surface approach for structural reliability problems. Structural Safety 7 (1), 57–66.

Cardaliaguet, P., Euvrand, G., 1992. Approximation of a function and its derivatives with a neural network. Neural Networks 5, 207–220.

Chapman, O.J.V., Crossland, A.D., 1995. Neural networks in probabilistic structural mechanics. In: Sundararajan, C. (Ed.), Probabilistic Structural Mechanics Handbook: Theory and Industrial Application. Chapman & Hall, pp. 317–330.

Chen, T., Chen, H., 1995. Approximation capability to functions of several variables, nonlinear functional and operators by radial basis function neural networks. IEEE Transactions on Neural Networks 32 (6), 904–9l0.

Christian, J.T., Baecher, G.B., 2002. The point-estimate method with large numbers of variables. International Journal for Numerical and Analytical Methods in Geomechanics 26 (15), 1515–1529.

Deng, J., Yue, Z.Q., Tham, L.G., Zhu, H.H., 2003. Pillar design by combining finite element methods, neural networks and reliability: a case study of the Feng Huangshan copper mine, China. International Journal of Rock Mechanics and Mining Sciences 40 (4), 585–599.

Deng, J., Gu, D.S., Li, X.B., Yue, Z.Q., 2005. Structural reliability analysis for implicit performance function using artificial neural network. Structural Safety 25 (1), 25–48.

Der Kiureghian, L., 1996. Structural reliability methods for seismic safety assessment: a review. Engineering Structures 18 (6), 412–424.

Ditlevsen, O., Madsen, H.O., 1996. Structural Reliability Methods. John Wiley & Sons, New York.

Faravelli, L., 1989. Response surface approach for reliability analysis. Journal of Engineering Mechanics 115 (12), 2763–2781.

Franke, R., 1982. Scattered data approximation: tests of some methods. Mathematics of Computation 38, 181–200.

Freudenthal, A.M., 1947. The safety of structures. Transactions, ASCE 112, 125–159.

Freudenthal, A.M., Garrelts, J.M., Shinozuka, M., 1966. The analysis of structural safety. Journal of the Structural Division, ASCE 92 (ST1), 267–325.

Garrett, J.H.J., 1994. Where and why artificial neural networks are applicable in civil engineering. Journal of Computing in Civil Engineering, ASCE 8 (2), 129–130.

Goh, A.T.C., 1994. Seisimic liquefaction potential assessed by neural networks. Journal of Geotechnical Engineering, ASCE 120 (9), 1467–1480.

Goh, A.T.C., Kulhawy, F.H., 2003. Neural network approach to model the limit state surface for reliability analysis. Canadian Geotechnical Journal 40 (6), 1235–1244.
Gomes, H.M., Awruch, A.M., 2004. Comparison of response surface and neural network with other methods for structural reliability analysis. Structural Safety 26 (1), 49–67.
Grigoriu, M., 2000. Stochastic mechanics. International Journal of Solids and Structures 37 (1–2), 197–214.
Guan, X.L., Melchers, R.E., 1997. Multitangent-plane surface method for reliability calculation. Journal of Engineering Mechanics 123 (10), 996–1002.
Haldar, A., Mahadevan, S., 2000. Reliability Assessment Using Stochastic Element Analysis. John Wiley & Sons, New York.
Harr, M.E., 1987. Reliability Based Design in Civil Engineering. McGraw-Hill, New York.
Hasofer, A.M., Lind, N.C., 1974. Exact and invariant second moment code format. Journal of the Engineering Mechanics Division, ASCE 100 (EM1), 111–121.
Haykin, S., 1999. Neural Networks: A Comprehensive Foundation, second ed. Prentice-Hall, New Jersey.
Hecht-Nielsen, R., 1989. Neurocomputing. Addison-Wesley Publishing Company, San Diego.
Hornik, K., Stinchcombe, M., White, H., 1989. Multi-layer feed-forward networks are universal approximators. Neural Networks 2, 359–368.
Hornik, K., Stinchcombe, M., White, H., 1990. Universal approximation of an unknown mapping and its derivatives using multi-layer feed-forward networks. Neural Networks 3, 551–560.
Hurtado, J.E., 2002. Analysis of one-dimensional stochastic finite elements using neural networks. Probabilistic Engineering Mechanics 17, 35–44.
Hussain, M.F., Barton, R.R., Joshi, S.B., 2002. Metamodeling: radial basis functions, versus polynomials. European Journal of Operational Research 138 (1), 142–154.
Kaymaz, I., 2005. Application of kriging method to structural reliability problems. Structural Safety 27 (2), 133–151.
Kim, S.H., Na, S.W, 1997. Response surface method using vector projected sampling points. Structural Safety 19 (1), 3–19.
Lawson, J., Erjavee, J., 2001. Modern Statistics for Engineering and Quality Improvement. Thompson Learning, Pacific Grove, CA, USA.
Li, X., 1996. Simultaneous approximations of multivariate functions and derivatives by neural networks with one hidden layer. Neurocomputing 12, 327–343.
Madsen, H.O., Krenk, S., Lind, N.C., 1986. Methods of Structural Safety. Prentice-Hall, Inc., Englewood Cliffs, New Jersey.
Mai-Duy, N., Tran-Cong, T., 2003. Approximation of function and its derivatives using radial basis function networks. Applied Mathematical Modelling 27 (3), 197–220.
Masters, T., 1993. Practical Neural Network Recipes in C++. Academic Press, San Diego.
McDonald, D.B., Grantham, W.J., Tabor, W.L., 2000. Response surface model development for global/local optimization using radial basis functions. In: 8th Symposium on Multidisciplinary Analysis and Optimization, 6–8, Long Beach,CA (AIAA 2000-4776).
Meckesheimer, M., 2001. A framework for metamodel-based design: subsystem metamodel assessment and implementation issues. Ph.D. Dissertation. The Pennsylvania State University, USA.
Melchers, R.E., 1990. Radial importance sampling for structural reliability. Journal of Engineering Mechanics, ASCE 116 (1), 189–203.
Melchers, R.E., 1999. Structural Reliability Analysis and Predictions, second ed. John Wiley & Sons, New York.
Moody, J., Darken, C.J., 1989. Fast learning in networks of locally tuned processing units. Neural Computation 1, 281–294.
Morgenstern, D., 1956. Einfache Beispiele Zweidimensionaler Verteilungen. Mitteilingsblatt fur Mathematische Statistik 8, 234–235.
Nataf, A., 1962. Determination des distribution don't les marges sont donees. Comptes Rendus de l'Academic des Sciences 225, 42–43.
Nowak, A.S., Collins, K.R., 2000. Reliability of Structures. McGraw-Hill Companies, Boston.
Orr, M.J.L., 1999. Recent Advances in Radial Basis Function Networks, Technical Report, University of Edinburgh.
Park, J., Sandberg, I., 1993. Approximation and radial-basis-function networks. Neural Computation 5, 305–316.
Press, W.H., Teukolsky, S.A., Vetterling, W.T., Flannery, B.P., 1992. Numerical Recipes in FORTRAN: The Art of Scientific Computing, second ed. Cambridge University Press.
Rackwitz, R., Fiessler, B., 1976. Note on discrete safety checking when using non-normal stochastic models for basic variables. Loads Project Working Session, MIT, Cambridge, MA, USA, pp. 489–494.
Rackwitz, R., Fiessler, B., 1978. Structural reliability under combined random load sequences. Computers and Structures 9 (5), 484–494.
Rahman, S., Rao, B.N., 2001. An element-free Galerkin method for probabilistic mechanics and reliability. International Journal of Solids and Structures 38, 9313–9330.
Rajashekhar, M.R., Ellingwood, B.R., 1993. A new look at the response surface approach for reliability analysis. Structural Safety 12 (1), 205–220.
Rosenblueth, E., 1975. Point estimates for probability moments. Proceedings of National Academy of Science of the United States of America 72 (10), 3812–3814.
Rosenblueth, E., 1981. Two-point estimates in probabilities. Applied Mathematical Modelling 5, 329–335.

Sasaki, T., 2001. A neural network-based response surface approach for computing failure probabilities. In: Corotis, R.B., Schuëller, G.I., Shinozuka, M. (Eds.), 8th International Conference on Structural Safety and Reliability (ICOSSAR2001), USA.

Schueremans, L., 2005. Use of Meta-Models in structural reliability—new issues in the applicability of probabilistic techniques for construction technology. Post-doctoral Research Report, KULeuven, Belgium.

Schueremans, L., Van Gemert, D., 2005. Benefit of splines and neural networks in simulation based structural reliability analysis. Structural Safety 27, 246–261.

Shao, S., Murotso, Y., 1997. Structural reliability analysis using neural networks. Japanese Society in Mechanical Engineering (JSME) International Journal, Series A 40 (3), 242–246.

Shinozuka, M., 1983. Basic analysis of structural safety. Journal of Structural Engineering, ASCE 109 (3), 721–740.

Warnes, R.M., Glassey, J., Montague, A.G., Kara, B., 1998. Application of radial basis function and feedforward artificial neural networks to the *Escherichia coli* fermentation process. Neurocomputing 20, 67–82.

Wong, F.S., 1985. Slope reliability and response surface method. Journal of Geotechnical Engineering Division, ASCE 111 (1), 32–53.

Zhao, G.F., 1996. Reliability Theory and its Applications for Engineering Structures. Dalian University of Technology Press, Dalian (in Chinese).