Structural Engineering and Mechanics, *Vol. 71, No. 4 (2019) 329-339*
DOI: https://doi.org/10.12989/sem.2019.71.4.329

An evolutionary approach for structural reliability

Alireza Garakaninezhad[1] and Morteza Bastami*[2]

[1]*Iranian Academic Center for Education, Culture and Research, Kerman 7616914111, Iran*
[2]*International Institute of Earthquake Engineering and Seismology (IIEES), No. 26, Arghavan St., North Dibajee, Farmanieh, P.O. Box: 19395/3913, Tehran, Iran*

(Received February 4, 2018, Revised March 17, 2019, Accepted April 1, 2019)

Abstract. Assessment of failure probability, especially for a complex structure, requires a considerable number of calls to the numerical model. Reliability methods have been developed to decrease the computational time. In this approach, the original numerical model is replaced by a surrogate model which is usually explicit and much faster to evaluate. The current paper proposed an efficient reliability method based on Monte Carlo simulation (MCS) and multi-gene genetic programming (MGGP) as a robust variant of genetic programming (GP). GP has been applied in different fields; however, its application to structural reliability has not been tested. The current study investigated the performance of MGGP as a surrogate model in structural reliability problems and compares it with other surrogate models. An adaptive Metropolis algorithm is utilized to obtain the training data with which to build the MGGP model. The failure probability is estimated by combining MCS and MGGP. The efficiency and accuracy of the proposed method were investigated with the help of five numerical examples.

Keywords: reliability; genetic programming; surrogate model; metropolis algorithm; failure probability; Monte Carlo

1. Introduction

Failure probability can be defined as:

$$p_f = \int_{h(x)\leq 0} f_x(x)\, dx \tag{1}$$

where $\boldsymbol{X} = (x_1, x_2, \dots, x_d)$ is the d-dimensional vector of random variables, $f_x(x)$ is the joint probability density function (PDF) of x and $h(x)$ is the limit state function. This integration is calculated over the failure domain where $h(x) \leq 0$. Evaluation of this multi-dimensional integral is difficult except for some limited problems, especially when the limit state function is a function of structural responses such as displacement and stress, which can be obtained through finite element (FE) analysis and are determined implicitly. Numerous methods have been proposed to estimate the probability of failure. These can be categorized as geometric approximations of the failure region, Monte Carlo simulation (MCS) and surrogate models (Li 2012).

The first-order reliability method (FORM) (Der Kiureghian, Lin *et al*. 1987, *Brei*tung and Hohenbichler 1989, Tvedt 1990, Goda and Atkinson 2010) and second-order reliability method (SORM) (Hasofer. and Lind 1974, Rackwitz and Flessler 1978, Liu and Der Kiureghian 1991) are common geometric approximation methods. FORM and SORM approximate the limit state function near the most probable point using a hyper-plane and paraboloid, respectively. It is evident that the performance of these methods strongly depends on how close the approximated failure region is to the exact failure region. In cases in which the performance function and the corresponding the failure region are highly nonlinear, the approximation is likely be inappropriate, which will decrease the accuracy of the methods. In addition, these methods cannot estimate the approximation error.

Monte Carlo simulation (MCS) is based on the law of large numbers and is applied to approximate the multiple integral in Eq. (1). The MCS approach is initiated by generating independent and identically-distributed samples according to distribution F_X. The performance function at each sample generated is then evaluated. Next, the failure probability is obtained as the ratio between the number of samples falling in the failure region and the total size of the sample set. Although MCS is a straightforward method, its main disadvantage is the formidable computational efforts required by the considerable number of sampling cycles.

Some techniques have been developed to reduce the number of samples without losing accuracy. Importance sampling (IS) (Robert and Casella, Shanmugam and Balaban 1980, Hohenbichler and Rackwitz 1988, Echard, Gayton *et al*. 2013), subset simulation (SS) (Au and Beck 2001, Ching, Au *et al*. 2005, Giovanis, Papaioannou *et al*. 2017, Alvarez, Uribe *et al*. 2018), and line sampling (Pradlwarter, Schueller *et al*. 2007, Valdebenito, Jensen *et al*. 2018) are all alternative sampling-based methods. Where the analysis of complex structures using a FE model is involved, however, the number of samples required is still considerable. To overcome the computational costs, one alternative way is to use surrogate models instead of evaluating the FE computer model. Surrogate models are built by fitting a set of observations of system responses. These models approximate the failure region using a simple and explicit function.

*Corresponding author, Associate Professor
E-mail: m.bastami@iiees.ac.ir

ISSN: 1225-4568 (Print), 1598-6217 (Online)

The response surface methods (RSMs) are the earlier surrogate models which are usually quadratic. Kmiecik and Soares (2002) used the RSM to determine the cumulative distribution function of the strength of compressed plates. Teixeira and Soares (2010) extended this approach for reliability problems involving a random field of corrosion. Gaspar, Naess *et al*. (2014) combined RSM with MCS to estimate the failure probability of complex structural systems. Fang and Tee (2017) combined response surface and genetic algorithm in structural reliability.

Artificial neural networks (ANNs) have been used as surrogate models to assess failure probability by Hurtado and Alvarez (2001). ANN estimates nonlinear mapping from the input to the output set of a system. Several studies have assessed the performance of ANN in the domain of reliability analysis (McCulloch and Pitts 1943, Hornik, Stinchcombe *et al*. 1989, Elhewy, Mesbahi *et al*. 2006, Bucher and Most 2008, Chojaczyk, Teixeira *et al*. 2015, Chojaczyk, Teixeira *et al*. 2015). It is clear that ANN provides more flexibility than RSM in this area; however, constructing an ANN for a complex system requires huge computational outlay for training algorithms.

Support vector machines (SVMs) as powerful statistical learning techniques have been used for structural reliability by Hurtado and Alvarez (2003). They combined a SVM as a classification approach and the stochastic FE method to analyze structural reliability. Li, Lü *et al*. (2006) proposed two SVM-based approaches, SVM-based FORM and SVM-based MCS, and compared their performance to those of conventional RSM and ANNs. Zhao, Ru *et al*. (2014) implemented SVM-based RSM in combination with FORM for tunnel reliability analysis. Tan, Bi *et al*. (2011) compared the SVM-based RSM with the radial basis function neural network-based RSM and showed that there was no significant difference between these methods. Dai, Zhang *et al*. (2012) proposed a local approximation approach based on support vector regression (a regression model used in SVM) and adaptive Markov chain simulation. The efficiency and accuracy of SVM as a surrogate model have been investigated by other researchers (Song, Choi *et al*. 2013, Liu, Wu *et al*. 2017, Pan and Dias 2017, Xiang, Li *et al*. 2017).

Kriging interpolation is a powerful prediction tool used to build a surrogate model to approximate the limit state function. This technique was introduced in geostatistics by Karig and Matheron (Matheron 1973). Sacks, W. J. Welch *et al*. (1989) applied this method to computer experiments to solve deterministic optimization problems. Kaymaz (2005) introduced kriging to estimate structural failure probability and compared it with RSMs. The accuracy of this model, as are those of the other surrogate models, is strongly dependent on the training points. If the points selected are close to the limit state function, the performance of the surrogate model improves. Kriging has been combined with different methods to improve its efficiency and accuracy (Huang, Chen *et al*. 2016). In addition, in problems with the nonlinear limit state function, the accuracy of surrogate models is very important and it can decrease the number of calls to the numerical model. Although the approaches proposed based on kriging and adaptive enrichment are efficient, the performance of method is strongly dependent on the method used to add new points. Therefore, more robust models are still required to accurately approximate the limit state function. This study proposed a method based on genetic programming as a surrogate model which has not been tested in structural reliability.

Genetic programming (GP) as an alternative predictive approach is proposed herein to compute failure probability. In general, GP is specialization of genetic algorithms (GAs); both are based on Darwinian natural selection (Koza 1992). Genetic operators used in GAs can be implemented in GP with a little change. The main difference between GA and GP is the presentation of solutions. In GP, the results are presented by computer programs while, in GA, a binary string is proposed for the solution. The programs generated by traditional GP are tree structures and are expressed using a functional programming language (Koza 1992, Gandomi, Alavi *et al*. 2010). The main advantage of a GP-based approach over other surrogate models is its ability to generate predictive equations without assuming the prescribed form of an existing relationship.

Different strategies have been proposed to improve the traditional GP. GP and its variants have been applied to different kinds of structural engineering problems (Parsons and Canfield 2002, Yeun, Kim *et al*. 2005, Gao, Xiao *et al*. 2012). Multi-gene genetic programming (MGGP) is a robust variant of GP in which an individual solution consists of a number of genes which are combined linearly.

The present paper is organized as follows. Section 2 introduces GP and MGGP. Section 3 describes the adaptive Metropolis sampling technique. The proposed method is introduced in section 4. Section 5 illustrates the efficiency and accuracy of the proposed method using five examples. Section 6 concludes the article.

2. Genetic programming and MGGP

2.1 Genetic programming

Genetic programming was introduced by Koza (1992) and is a symbolic regression technique that solves problems without having a prescribed form for the solution. Unlike surrogate models such as ANN, GP is a self-parametrizing method that constructs the model without user training. GP is an extension of GA; most genetic operators applied in GA can be developed and implemented in GP. Although these methods are similar, the main difference between them relates to the representation of the solutions. In GP, individual solutions are computer programs that are usually represented as parse trees. In GA, a string of numbers is generated as the solution (Koza 1992).

Each individual in GP is represented as a tree structure comprising terminals and functions. The terminals and functions are selected from a pre-defined set in the GP system. The function set consists of useful arithmetic operators (+, -, ×, and ÷), mathematical functions (cos(), sin(), tan(), etc.), Boolean operators (And, Not, Or), logical expression (If, Then) or any other reasonable function defined by the user. The terminal set may consist of the

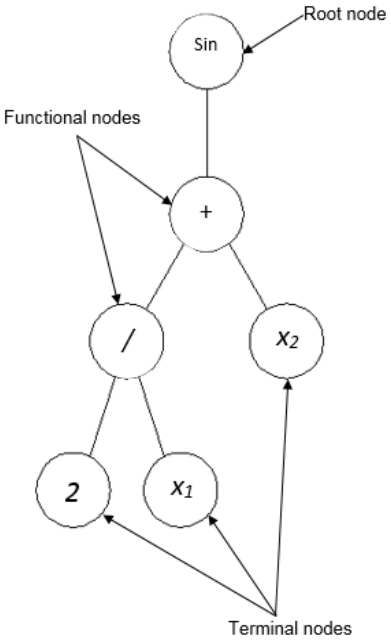


Fig. 1 The tree representation of a GP model

variables and constants of the programs (Gandomi, Alavi *et al*. 2010). A set of functions and terminals are randomly selected and combined to form a computer model. This model is a tree-like structure which consists of a root node with branches extending from each function and ending in a terminal node. An example of a computer model is shown in Fig. 1.

After a set of GP trees is generated at random, the fitness value is calculated for each individual using the objective function. A set of computer models are selected based on their fitness value for reproduction. Then a new population is created by implementing the genetic operators. Crossover and mutation are the basic genetic operations. For a crossover operator, given two parents as shown in Fig. 2, a crossover point in each parent tree is randomly selected. The offspring is then created by replacing the sub-tree rooted at the crossover point in a copy of the first parent along with the sub-tree rooted at the crossover point of the second parent. For mutation operators, a function or terminal is randomly selected and mutated. The GP continues replacing the old generation with the new one. This process is repeated until one of the stopping conditions is satisfied. The individual which has the best fitness value is selected as the result of the GP (Koza 1992).

2.2 Multi-gene genetic programming (MGGP)

Symbolic regression is generally implemented using a traditional GP to evolve a population of trees. Each tree predicts a ($n\times1$) vector of output y through a corresponding ($n\times d$) matrix of input X, in which n is the number of observations and d is the number of variables (Searson, Leahy *et al*. 2010). In MGGP, a symbolic model consists of a weighted linear combination of GP trees, also called genes. Fig. 3 shows a typical multi-gene model. This model predicts one output variable using three input variables (x_1, x_2, and x_3). This model configuration includes a nonlinear term (e.g. tan) but is linear in the parameters with respect to coefficients b_0, b_1, and b_2. In MGGP, the maximum number of genes in a model (G_{max}) and the maximum tree depth for any gene are the main parameters used to construct a model. These parameters control the maximum complexity of the evolved models. In particular, the restriction of tree depth usually leads to the evolution of a relatively compact model.

The optimal linear coefficients for each model are evaluated using the training population and ordinary least squares methods. MGGP is an efficient tool that can combine the power of classical linear regression with the ability to capture the nonlinear behavior of a system. A comparison of MGGP and traditional GP shows that MGGP is more accurate and computationally efficient than the standard GP for symbolic regression (Searson, Willis *et al*. 2007, Searson, Leahy *et al*. 2010).

The initial population in MGGP is obtained by creating individuals, including GP trees, which are randomly generated with different genes. MGGP, in addition to the standard GP recombination operators, uses a crossover mechanism called the two-point high-level crossover operator which allows the exchange of genes between individuals. During a MGGP run, implementing this operator may delete some genes.

Let G_i be the i^{th} gene of an individual. The first parent contains three genes (G_1, G_2, G_3) and the second contains four genes (G_4, G_5, G_6, G_7). For each individual, two randomly-selected crossover points are generated. The genes enclosed by a crossover point are exchanged. It is assumed that G_2 in the first parent and G_5 and G_6 in the second are enclosed by the crossover points. The enclosed genes are exchanged and the new individuals result as follows:

$$(G_1, G_5, G_6, G_3) \quad (G_4, G_2, G_7).$$

This combination operator allows acquisition of new genes for both individuals. Some genes can be removed by applying this combination operator. When an exchange of genes creates new ones with more genes than G_{max}, the genes are randomly selected and removed until the number of genes in an individual decreases to less than G_{max}. Standard subtree crossover can be applied in the MGGP algorithm. This operator is known as a low level crossover in which a gene is selected at random from each parent. The standard subtree crossover is then applied and the constructed trees replace the parent trees in the otherwise unchanged individual in the next generation.

Mutation can be performed in MGGP using the following methods: (1) sub-tree mutation; (2) mutation of constants using an additive Gaussian perturbation; (3) setting a randomly-selected constant to zero s; (4) substitution of a randomly-selected input node by another randomly-selected input node; (5) setting a randomly-selected constant to one and; (6) substitution of a randomly-selected constant with another randomly generated constant (Searson, Leahy *et al*. 2010). The probability of each evolutionary process can be specified by the user. These

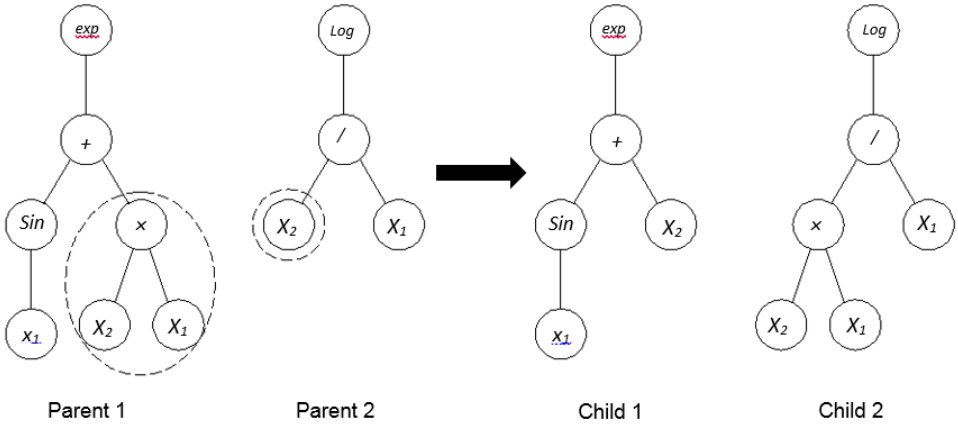
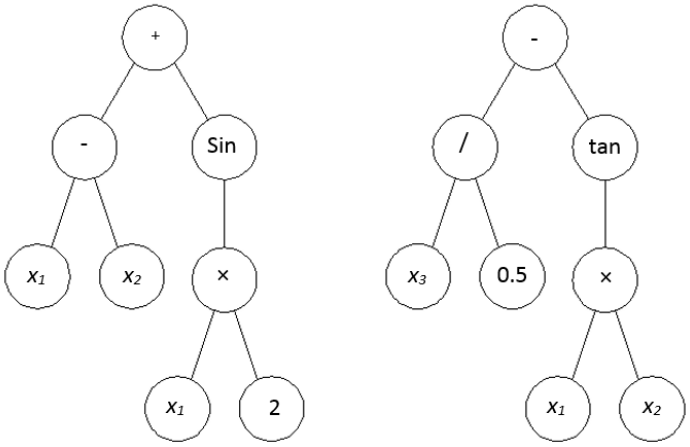


Fig. 2 Cross-over operation in GP



$$y = b_0 + b_1\left((x_1 - x_2) + sin\ (2x_1)\right) + b_2\left(\frac{x_3}{0.5} - tan(x_1 x_2)\right)$$

Fig. 3 A typical example of multi-gene GP model

processes are grouped into categories which are called events. Their probabilities must sum to one.

3. Generation of dataset using adaptive Metropolis algorithm

The first step in the proposed method is generation of samples from the region of interest. Because the failure region is not a closed-form expression, traditional MCS cannot be handled directly. Hence, adaptive sampling approaches have been received more attention these years. A review and discussion of these approaches have been presented by Liu, Ong *et al*. (2017). In this situation, a Markov chain simulation using the Metropolis algorithm can be implemented to achieve samples having a favored target distribution (Au and Beck 2001, Au and Beck 2003, Yuan, Lu *et al*. 2013). The classic metropolis algorithm is non-adaptive, therefore the target PDF at each iteration will be unchanged. In some cases, its performance may not be good and the convergence of the generated samples to target distribution may be delayed. Adaptive algorithms that use all samples in the chain to tune the proposal distribution have been proposed to overcome these difficulties.

The adaptive Metropolis algorithm is formed on the basis of the classic random-walk Metropolis algorithm (Metropolis, Rosenbluth *et al*. 1953). Suppose π(x) is a target distribution function and is sampled in the current step in set $(x_0, x_1, \dots, x_t)$. Candidate point x^* is generated from proposal density function $q(.|(x_0, x_1, \dots, x_t))$ which depends on all samples in the chain. This point is accepted as the sample point of the next step with probability of:

$$r = min\left(1, \frac{\pi(x^*)}{\pi(x_t)}\right) \quad (2)$$

The candidate point is rejected with the remaining probability of $(1 - r)$. In this case, set $x_{t+1} = x_t$.

The proposal distribution used in this method can be selected arbitrarily, but the convergence rate of the generated chain will accelerate if this distribution approaches the true target distribution. Common choices for the proposal distribution have a multivariate uniform or multivariate normal distribution. It has been shown that the results obtained with a uniform distribution can be sensitive with respect to the parameters. In the present study, the

multivariate normal distribution was employed. The proposed distribution is centered at the current state of the samples, x_t, and covariance matrix $C_t = C_t(x_0, x_1, \dots, x_t)$. The challenging issue regarding the adaption is how the covariance of the proposed distribution depends on the entire chain. The covariance matrix affects the deviation of the next point from the actual point and plays an important role in the performance of the Metropolis algorithm. Haario, Saksman *et al*. (2001) proposed a covariance matrix as follows:

$$C_t = s_d\, cov(x_0, x_1, \dots, x_t) + s_d\, \varepsilon\, \boldsymbol{I_d} \tag{3}$$

where d is the dimension of the random variable, s_d is a scaling parameter and is proposed to be a basic approximate as $s_d = 2.4^2/d$, ε is a positive constant that can be very small and $\boldsymbol{I_d}$ is a d-dimensional identity matrix. In general, the definition of the empirical covariance matrix for the samples $(x_0, x_1, \dots, x_t)$ is:

$$cov(x_0, x_1, \dots, x_k) = \frac{1}{k}\left(\sum_{i=0}^{k} x_i x_i^T - (1+k)\bar{x}_k \bar{x}_k^T\right) \tag{4}$$

where $\bar{x}_k = 1/(1+k)\sum_{i=0}^{k} x_i$, which can be obtained using Eq. (3). The covariance matrix at step $t+1$ (C_{t+1}) satisfies the following recursion equation as:

$$C_{t+1} = \frac{t-1}{t} C_t + \frac{s_d}{t}(t\, \bar{\boldsymbol{X}}_{t-1}\, \bar{\boldsymbol{X}}_{t-1}^T - (t+1)\, \bar{\boldsymbol{X}}_{t-1}\, \bar{\boldsymbol{X}}_{t-1}^T) + \varepsilon\, \boldsymbol{I_d} \tag{5}$$

Covariance matrix C_{t+1} can be calculated at small computational cost. Because the adaptive Metropolis algorithm generates the candidate point according to the overall history, this algorithm is non-Markovian. Haario *et al*. (2001) showed that this algorithm has the correct ergodic properties and can converge to correct the target distribution.

4. Proposed method

In this section, the proposed method to evaluate the failure of probability is describe. MGGP is used as a surrogate model to approximate the exact limit state function. The method employs adaptive Metropolis algorithm to generate the training population. The main steps of the proposed method are as follows:

1. In the first step of implementation of GP, n input-output pairs are required as the initial population. These samples are randomly generated according to the PDF of input variable $\boldsymbol{x}$ and their limit state function is evaluated.
2. The sample set is then divided into the learning and validation. The learning subset is used to train the MGGP model. The validation subset is used to evaluate the accuracy of the models generated for sampling which are not used for training. In order to achieve a reliable MGGP model, several combinations of training and validation data set are employed. This step aims to determine the main parameters of MGGP. The number of learning data set can be considered as 80% of population and 20% can be used for validating. After determining the parameters, both the learning and validation subsets are used for construction of the MGGP model and are called training data.
3. Computation of an MGGP model based on the training data using the GPTIPS toolbox (Searson, Leahy *et al*. 2010). This open-source toolbox is widely used for GP reference efforts.
4. Evaluation of the performance of the MGGP constructed in step 3 is carried out using the correlation coefficient (R), mean absolute error (MAE) and root mean square error (RMSE). R, MAE, and RMSE are defined as follows:

$$R = \frac{\sum_{i=1}^{N}(T_i - \bar{T}_i)(P_i - \bar{P}_i)}{\sqrt{\sum_{i=1}^{N}(T_i - \bar{T}_i)^2 \sum_{i=1}^{N}(P_i - \bar{P}_i)^2}} \tag{6}$$

$$MAE = \frac{\sum_{i=1}^{N}|T_i - P_i|}{N} \tag{7}$$

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}|T_i - P_i|^2}{N}} \tag{8}$$

where T_i and P_i are the observed and predicted values, respectively, and N is the number of samples.

5. After constructing the MGGP model, the failure probability can be estimated using MCS or any simulation method with a variance reduction technique. First, the MCS sample set $i = 1,2, \dots, n_{MCS}$ is generated. Next, using GPTIPS, the values of the limit state function at the samples are evaluated. The probability of failure can then be calculated as follows:

$$p_f \cong \hat{p}_f = \frac{n_{H\leq 0}}{n_{MCS}} \tag{9}$$

where $n_{H\leq 0}$ is the number of samples with limit state functions that are null or negative.

6. The initial population set and MGGP model are updated. A new point, x^*, is generated using the adaptive Metropolis algorithm and the true limit state function on the generated point is calculated. The distribution is considered as multivariate normal distribution with center at the current state of the samples, x_t. Next, return to step 2.
7. Repeat steps 2-6 until the failure probability obtained satisfies convergence as follows:

$$\left|\frac{\mathrm{p_f^{(k)}} - \mathrm{p_f^{(k-1)}}}{\mathrm{p_f^{(k)}}}\right| < \varepsilon \tag{10}$$

where $p_f^{(k)}$ is the present estimation and $p_f^{(k-1)}$ is the previous value of failure probability. Threshold ε can be chosen as a value between 10^{-4} and 10^{-1}.

Table 1 Proposed ranges of parameters used in MGGP algorithm

parameter	ranges
Population size	200-1000
Number of generations	500-1000
Maximum number of genes allowed in a model	8-20
Maximum tree depth	3-6
Tournament size	10% of population size
Crossover probability	0.7-0.9
Low-level crossover	0.6-0.9
High-level crossover	0.15-0.5
Mutation probability	0.1-0.2
Sub-tree mutation	0.5-0.9
Substituting input terminal with another random terminal	0.05
Gaussian perturbation of randomly selected constant	0.05
Ephemeral random constant	[-10 10]

4.1 Implementation issues

GPTIPS toolbox and other subroutines in MATLAB are used to estimate the failure probability by MGGP. Different parameters are involved in MGGP applications. The number of generations, population size, and maximum allowable genes in each individual, and maximum tree depth play important roles in accurately constructing a MGGP model. The number of generations determines the number of levels run in MGGP before termination and the number of models generated is determined by the population size. In general, an increase in these parameters increases the accuracy of the model. A MGGP model with high parameter values, especially for G_{max}, can more accurately predict a nonlinear complex system; however, the risk of over-fitting may increase.

To avert over-fitting in the proposed method, a testing subset is used. GPTIPS includes a number of mechanisms intended to mitigate this; however, in early iterations where the number of samples are few relative to the number of variables, it is strongly recommended to assign a moderate number of generations. Configuration parameters exist in addition to these parameters. Table 1 lists the parameters used in MGGP and their proposed ranges. These parameters can be assigned by considering the complexity of problem and after trial-and-error. To obtain consistent data division, it is better to combine several training and testing subsets; however, in this study, these subsets were considered to be unchanged during the running of the algorithm. Of the total data, 80% were applied for training and the rest data for testing.

5. Numerical examples

To illustrate the performance of the proposed method, its results were compared with the results of five examples from previous studies of other surrogate models. The first three examples were chosen to illustrate method efficiency and each have two random variables. The forth example is a system with a moderate number of non-random variables. The last example is a truss structure with an implicit limit state function.

5.1 Example 1: Two random variable problem

The first example featured an explicit limit state function with two random variables. This example was tested by Echard, Gayton *et al*. (2013). The limit state function is as follows:

$$H(x_1, x_2) = 0.5(x_1 - 2)^2 - 1.5(x_2 - 5)^3 - 3$$

where x_1 and x_2 are assumed to have a standard normal distribution.

The failure probability obtained by MCS with 10^8 samples was 2.873×10^{-5}. In the proposed method the MGGP surrogate model was constructed with a total of 20 calls, 15 as the step number of the Metropolis algorithm as the initial population and 5 as the number of added samples required by the proposed method. The starting point and samples generated by the adaptive Metropolis algorithm are shown in Fig. 4.

These samples were evaluated at the origin limit state function and were divided into training and testing subsets. The training subset was used to learn and construct the MGGP model using GPTIPS. The parameters required to define the MGGP structure were in range determined by Table 1.

In this example, the number of generations was 200, population size was 200 and G_{max} was 8. Other parameters were assigned as default values in GPTIPS. The accuracy of the model was evaluated by calculating R, MAE and RMSE using the testing subset. To investigate the effect of G_{max} on the accuracy of the MGGP model, four configurations at $G_{max} = 2, 4, 6$ and 8 were trained based on the aforementioned samples.

In order to investigate the effect of parameters, the normalized failure probability is defined as the ratio of exact P_f which is obtained from MCS to P_f obtained from the proposed method. Fig. 5 shows the normalized failure probability obtained by each model. It can be seen that the accuracy increased as G_{max} increased. When the value for G_{max} was low, it was necessary to increase the number of generations and population size to construct a model offering reasonable accuracy. The results are summarized in Table 2. As seen, the number of calls to the origin limit state function, N_{call}, was less than for the other surrogate models and the predicted failure probability was very accurate.

An important method for evaluation of the capabilities of the surrogate models is to determine whether or not the predicted values are close to the limit state function. Fig. 6 shows the exact limit state function and the predicted values using the MGGP model. As seen, the predicted values are very accurate.

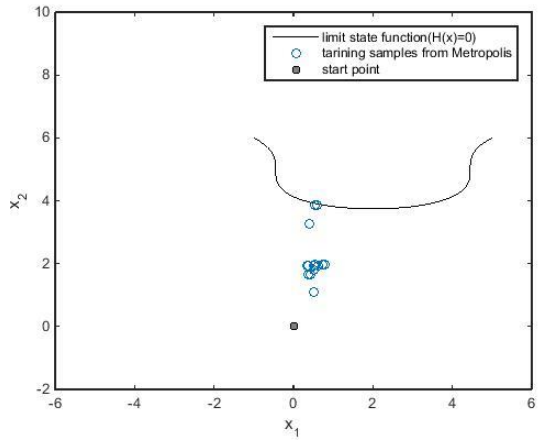


Fig. 4 The training subset generated by the proposed method

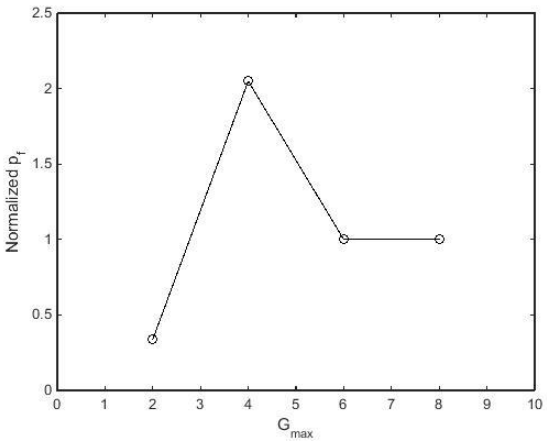


Fig. 5 The effect of G_{max}

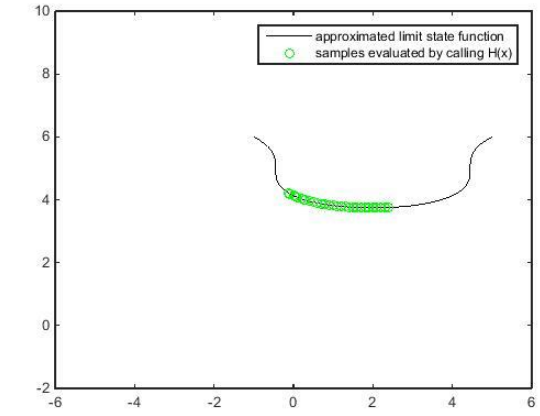


Fig. 6 The exact limit state function and the predicted value in Example 1

Table 2 Results of failure probability in Example 1

Method	N_{call}	P_f
MCS	10^8	2.873×10^{-5}
Proposed method	15+5	2.873×10^{-5}
Adaptive Kriging + IS	19+7	2.86×10^{-5}
IS	$19+10^4$	2.86×10^{-5}
FORM	19	4.21×10^{-5}

Table 3 Failure probability, Example 2

Method	N_{call}	P_f
MCS	10^8	8.9×10^{-5}
Proposed method	130+22	8.8×10^{-5}
Au and Beck	300+500	6.71×10^{-5}
Dai *et al.*	500+500	7.21×10^{-5}
Changcong *et al.*	106+642	7.89×10^{-5}

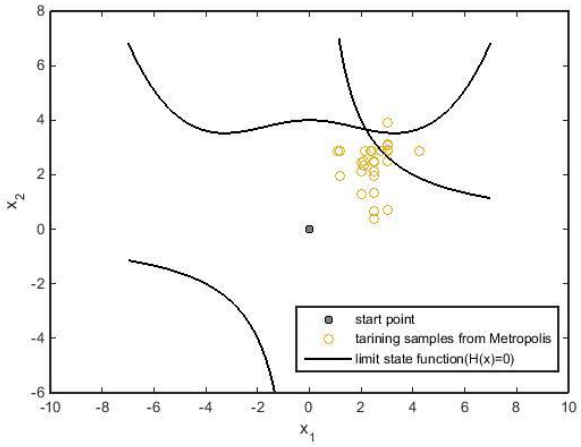


Fig. 7 The training samples in Example 2

5.2 Example 2: Series system with two design points

The second example is a series system using two explicit performance functions. This example has also been investigated in (Au and Beck 1999, Dai, Zhang *et al*. 2012, Changcong, Zhenzhou *et al*. 2015). This system has two standard normal distributed variables, x_1 and x_2, and the limit state function is defined as:

$$H_1(x_1,x_2) = 3 - x_2 + exp\left(-\frac{x_1^2}{10}\right) + \left(\frac{x_1}{5}\right)^4$$

$$H_2(x_1,x_2) = 8 - x_1\, x_2$$

$$H(x_1,x_2) = \min\,(H_1,H_2)$$

This performance function is illustrated in Fig. 7. The design points were (0, 4) and (2.83, 2.83). Comparison of the results from the examples and those obtained by the proposed method are shown in Table 3.

The failure probability estimated by MCS was 8.9×10^{-5} from 10^8 samples. The results show that the proposed method was more efficient than the other surrogate models. The predicted failure probability was also more accurate than the others. This accuracy was achieved by a lower number of calls to the origin limit state function.

5.3 Example 3: Four branch series system

This example is a series system with two random variables and four branches. It was studied in (Waarts 2000, Schueremans and Van Gemert 2005, Schueremans and Van Gemert 2005, Echard, Gayton *et al*. 2011) and is defined as:

$$H(x_1,x_2)=min\begin{Bmatrix}3+0.1(x_1-x_2)^2-\dfrac{\sqrt{2}(x_1 2x_2)}{2}\\3+0.1(x_1-x_2)^2+\dfrac{\sqrt{2}(x_1 2x_2)}{2}\\(x_1-x_2)+3\sqrt{2}\\(x_2-x_1)+3\sqrt{2}\end{Bmatrix}$$

where x_1 and x_2 are the standard normal distributed random variables. The results are compiled in Table 4.

The MGGP model was constructed based on a total of 57 calls, as shown in Fig. 8, among which 40 points were used as the initial population generated by adaptive Metropolis and 17 points were added during implementation of the proposed method. The results show that the proposed method was more efficient than the other surrogate models. The number of calls required to the origin limit state function was lower than for most of the other surrogate models, but the failure probability was very accurate.

Table 4 Results of failure probability in Example 3

Method	N_{call}	β	P_f
MCS	10^6	2.618	0.004416
Proposed method	75	2.618	0.004416
Adaptive Kriging-MCS+U	126	2.618	0.004416
Adaptive Kriging-MCS +EFF	124	2.619	0.004412
Directional Sampling (DS)	52	2.61	0.0045
DS + Response Surface	1745	2.57	0.0050
DS + Spline	145	2.82	0.0024
DS + Neural Network	165	2.64	0.0041
Importance Sampling (IS)	1469	2.58	0.0049
IS + Response Surface	1375	2.62	0.0045
IS + Spline	428	2.62	0.0045
IS + Neural Network	52	2.53	0.0057

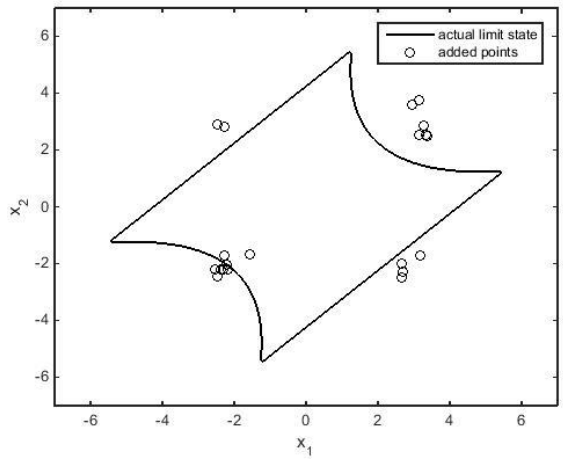


Fig. 8 Points generated during implementation of the proposed method

Table 5 Results of failure probability for Example 4

Method	N_{call}	P_f
MCS	5×10^6	**4.83×10^{-3}**
Proposed method	55+16	**4.83×10^{-3}**
Ziha	2500×4	**4.87×10^{-3}**
Yuan *et al*.	200×10	**4.53×10^{-3}**
Changcong *et al*.	95+337	**5.01×10^{-3}**

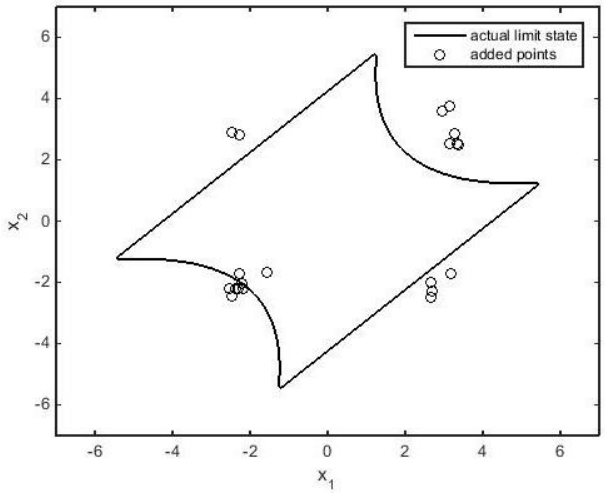


Fig. 9 Convergence of failure probability in Example 4

5.4 Example 4: System with non-normal variables

A plane frame structure, which is a series system with 7 non-normal variables, was considered. The following limit state functions was developed based on virtual work as:

$$\begin{cases}Frame: H_1(\boldsymbol{X})=x_1+2\,x_3+2\,x_4+\,x_5-h\,x_6-h\,x_7\\Beam: H_2(\boldsymbol{X})=x_2+2\,x_3+x_4+\,x_5-h\,x_7\\Sway: H_3(\boldsymbol{X})=x_1+x_2+x_4+\,x_5-h\,x_6\\H(\boldsymbol{X})=\min\left(H_1(\boldsymbol{X}),H_2(\boldsymbol{X}),\;H_3(\boldsymbol{X})\right)\end{cases}$$

where x_1-x_5 are the plastic moment capacities and x_6 and x_7 are the loads, which are log normally distributed independent variables with the following mean values and standard deviations: $E(x_i)=134.9, D(x_i)=13.49$, $i=1,2,3,4,5,\ E(x_6)=50, D(x_6)=15$ and $E(x_7)=40,\ D(x_7)=12$. Parameter h equalled 5.0. This example has been studied in (Žiha 1995, Yuan, Lu *et al*. 2013, Changcong, Zhenzhou *et al*. 2015).

The results of these studies were compared with those from the proposed method in Table 5. The results of Yuan show good accuracy, but considering the number of samples, the computational cost was high. The estimate obtained from the proposed method was more accurate and efficient than the others. This estimate was obtained with only 71 samples, 55 as the initial population and 16 added by the adaptive Metropolis algorithm. Fig. 9 shows the number of samples versus the failure probability and indicates that the estimate converged after 71 samples. This example shows the efficiency of the proposed method in problems with non-normal variables.

Table 6 Results of failure probability for Example 5

Method	N_{call}	P_f	*Error (%)*
MCS	10^6	**0.1394**	-
Proposed method	175	**0.1390**	**0.3**
FORM	127	**0.0862**	**38.2**
Rahman and Wei	187	0.1357	2.6

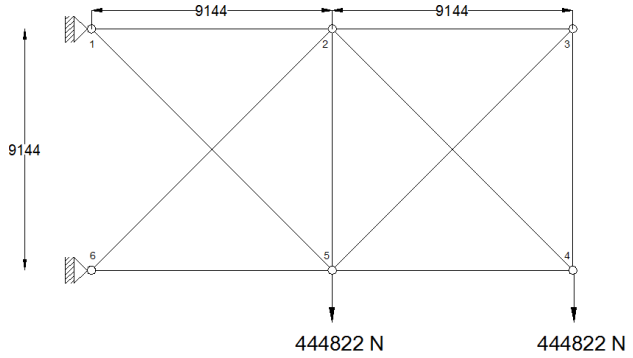


Fig. 10 Example 5, 10-Bar truss structure

5.5 Example 5: 2D truss structure

This example deals with a 2D truss structure composed of 10-bar elements and is shown in Fig. 10. It was selected to examine the performance of the proposed method for problems with an implicit limit state function. The structure layout, member sizes and Young's modulus are based on Rahman and Wei (2006). The random variables were the cross-sectional areas of members A_i, i=1,2,…,10. It was assumed that A_i is a normally-distributed variable with a mean and standard deviation of 16.13×10^{-4} m^2 and 3.23×10^{-4} m^2, respectively. The material Young's modulus was deterministic and equal to 68.96 GP. The limit state of interest was node 4 displacement v_4 with a limit value of 457.2 mm. It is expressed as:

$$H(x) = 457.2 - v_4$$

The limit state function is implicit and must be evaluated using FE analysis. It was assumed that the behavior of the model was linear elastic.

The failure probability obtained by MCS based on 10^6 samples was 0.1394. Fig. 11 shows the convergence of the failure probability as the size of the dataset samples increases. As seen, the MGGP model was trained using a small number of samples with good accuracy and a lower number of samples than the other methods (Table 6).

6. Conclusion

This paper introduced a method to estimate the failure probability of complex systems which is based on multi-gene genetic programming (MGGP) as a robust variant of genetic programming (GP) and Monte Carlo simulation. The main advantage of GP is that it is self-parametrizing, so constructing the model structure can occur without user training, unlike other surrogate models.

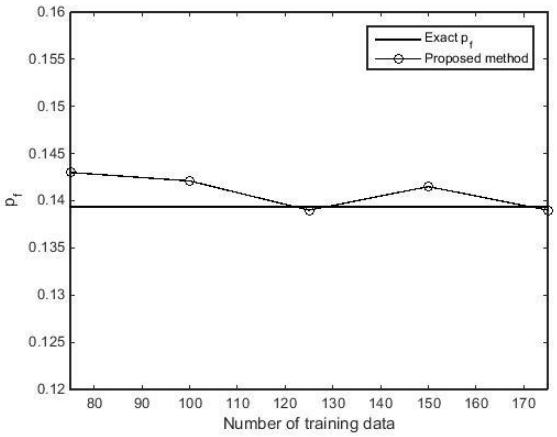


Fig. 11 Example 5, failure probability

This methodology begins with the generation of initial samples by the adaptive Metropolis algorithm which are then divided into training and testing subsamples. A MGGP model is constructed based on training samples and its accuracy is evaluated by testing samples to reduce the risk of over-fitting. The MGGP model is an explicit surrogate model and an approximation of the original limit state function or FE-based structures. Construction of a MGGP model requires parameters, four of which play important roles. Although these parameters can be obtained through trial-and-error, it has been shown that convergence of the proposed method does not depend on them.

Five examples were investigated to illustrate the efficiency and accuracy of the proposed method. The results show that the models having high MGGP parameters, especially for maximum allowable genes, in each individual were more accurate; however, the models were implemented with parameters in the lower range with reasonable computational cost. The results were compared with other surrogate models and MCS as the exact response. They indicate that the proposed method is very accurate and the number of samples required for evaluation of the true limit state function is reduced.

Acknowledgements

The authors would like to acknowledge International Institute Earthquake Engineering and Seismology (IIEES) for its help in providing research documents and Grant Number 659.

References

Alvarez, D.A., Uribe, F. and Hurtado, J.E. (2018), "Estimation of the lower and upper bounds on the probability of failure using subset simulation and random set theory", *Mech. Syst. Signal Process*, **100**, 782-801. https://doi.org/10.1016/j.ymssp.2017.07.040.

Au, S.K. and Beck, J.L. (1999), "A new adaptive importance sampling scheme for reliability calculations", *Struct. Safety*, **21**(2), 135-158. https://doi.org/10.1016/S0167-4730(99)00014-4.

Au, S.K. and Beck, J.L. (2001), "Estimation of small failure probabilities in high dimensions by subset simulation", *Probabilistic Eng. Mech.*, **16**(4), 263-277. https://doi.org/10.1016/S0266-8920(01)00019-4.

Au, S.K. and Beck, J.L. (2003), "Subset simulation and its application to seismic risk based on dynamic analysis", *J. Eng. Mech.*, **129**(8), 901-917. https://doi.org/10.1061/(ASCE)0733-9399(2003)129:8(901).

Breitung, K. and Hohenbichler, M. (1989), "Asymptotic approximations for multivariate integrals with an application to multinormal probabilities", *J. Multivariate Anal.*, **30**(1), 80-97. https://doi.org/10.1016/0047-259X(89)90089-4.

Bucher, C. and Most, T. (2008), "A comparison of approximate response functions in structural reliability analysis", *Probabilistic Eng. Mech.*, **23**(2), 154-163. https://doi.org/10.1016/j.probengmech.2007.12.022.

Changcong, Z., Zhenzhou, L., Feng, Z. and Zhufeng, Y. (2015), "An adaptive reliability method combining relevance vector machine and importance sampling", *Struct. Multidisciplinary Optimization*, **52**(5), 945-957. https://doi.org/10.1007/s00158-015-1287-z.

Ching, J., Au, S.K. and Beck, J.L. (2005), "Reliability estimation for dynamical systems subject to stochastic excitation using subset simulation with splitting", *Comput. Method Appl. Mech. Eng.*, **194**(12–16), 1557-1579. https://doi.org/10.1016/j.cma.2004.05.028.

Chojaczyk, A., Teixeira, A., Neves, L., Cardoso, J. and Soares, C.G. (2015), "Review and application of Artificial Neural Networks models in reliability analysis of steel structures", *Struct. Safety*, **52**, 78-89. https://doi.org/10.1016/j.strusafe.2014.09.002.

Chojaczyk, A., Teixeira, A., Neves, L.C., Cardoso, J. and Soares, C.G. (2015), "Review and application of artificial neural networks models in reliability analysis of steel structures", *Struct. Safety*, **52**, 78-89. https://doi.org/10.1016/j.strusafe.2014.09.002.

Dai, H., Zhang, H., Wang, W. and Xue, G. (2012), "Structural Reliability Assessment by Local Approximation of Limit State Functions Using Adaptive Markov Chain Simulation and Support Vector Regression", *Comput. Aid Civil Infrastruct. Eng.*, **27**(9), 676-686. https://doi.org/10.1111/j.1467-8667.2012.00767.x.

Der Kiureghian, A., Lin, H. and Hwang, S. (1987), "Second-Order Reliability Approximations", *J. Eng. Mech.*, **113**(8), 1208-1225. https://doi.org/10.1061/(ASCE)0733-9399(1987)113:8(1208).

Echard, B., Gayton, N. and Lemaire, M. (2011), "AK-MCS: an active learning reliability method combining Kriging and Monte Carlo simulation", *Struct. Safety*, **33**(2), 145-154. https://doi.org/10.1016/j.strusafe.2011.01.002.

Echard, B., Gayton, N., Lemaire, M. and Relun, N. (2013), "A combined importance sampling and kriging reliability method for small failure probabilities with time-demanding numerical models", *Raliability Eng. Syst. Safety*, **111**, 232-240. https://doi.org/10.1016/j.ress.2012.10.008.

Elhewy, A.H., Mesbahi, E. and Pu, Y. (2006), "Reliability analysis of structures using neural network method", *Probabilistic Eng. Mech.*, **21**(1), 44-53. https://doi.org/10.1016/j.probengmech.2005.07.002.

Fang, Y. and Tee, K.F. (2017), "Structural reliability analysis using response surface method with improved genetic algorithm", *Struct. Eng. Mech.*, **62**(2), 139-142. https://doi.org/10.12989/sem.2017.62.2.139.

Gandomi, A.H., Alavi, A.H., Arjmandi, P., Aghaeifar, A. and Seyednour, R. (2010), "Genetic programming and orthogonal least squares: a hybrid approach to modeling the compressive strength of CFRP-confined concrete cylinders", *J. Mech. Mater. Struct.*, **5**(5), 735-753. http://dx.doi.org/10.2140/jomms.2010.5.735.

Gao, L., Xiao, M., Shao, X., Jiang, P., Nie, L. and Qiu, H. (2012), "Analysis of gene expression programming for approximation in engineering design", *Struct. Multidisciplinary Optimization*, **46**(3), 399-413. https://doi.org/10.1007/s00158-012-0767-7.

Gaspar, B., Naess, A., Leira, B.J. and Soares, C.G. (2014), "System reliability analysis by Monte Carlo based method and finite element structural models", *J. Offshore Mech. Arctic Eng.*, **136**(3), 031603-031603. https://doi.org/10.1115/1.4025871.

Giovanis, D.G., Papaioannou, I., Straub, D. and Papadopoulos, V. (2017), "Bayesian updating with subset simulation using artificial neural networks", *Comput. Method Appl. Mech. Eng.*, **319**, 124-145. https://doi.org/10.1016/j.cma.2017.02.025.

Goda, K. and Atkinson, G.M. (2010), "Intraevent spatial correlation of ground-motion parameters using SK-net data", *Bullet. Seismologic. Soc. America*, **100**(6), 3055-3067. https://doi.org/10.1785/0120100031.

Haario, H., Saksman, E. and Tamminen, J. (2001), "An adaptive Metropolis algorithm", *Bernoulli*, 223-242.

Hasofer., A.M. and Lind, M.C. (1974), "An exact and invariant first order reliability format", *J. Eng. Mech.*, **100**, 111-121.

Hohenbichler, M. and Rackwitz, R. (1988), "Improvement of second-order reliability estimates by importance sampling", *J. Eng. Mech.*, **114**(12), 2195-2199. https://doi.org/10.1061/(ASCE)0733-9399(1988)114:12(2195).

Hornik, K., Stinchcombe, M. and White, H. (1989), "Multilayer feedforward networks are universal approximators", *Neural Networks*, **2**(5), 359-366. https://doi.org/10.1016/0893-6080(89)90020-8.

Huang, X., Chen, J. and Zhu, H. (2016), "Assessing small failure probabilities by AK–SS: an active learning method combining Kriging and subset simulation", *Struct. Safety*, **59**, 86-95. https://doi.org/10.1016/j.strusafe.2015.12.003.

Hurtado, J. and Alvarez, D. (2003), "Classification Approach for Reliability Analysis with Stochastic Finite-Element Modeling", *J. Struct. Eng.*, **129**(8), 1141-1149. https://doi.org/10.1061/(ASCE)0733-9445(2003)129:8(1141).

Hurtado, J.E. and Alvarez, D.A. (2001), "Neural-network-based reliability analysis: a comparative study", *Comput. Method Appl. Mech. Eng.*, **191**(1–2), 113-132. https://doi.org/10.1016/S0045-7825(01)00248-1.

Kaymaz, I. (2005), "Application of kriging method to structural reliability problems", *Struct. Safety*, **27**(2), 133-151. https://doi.org/10.1016/j.strusafe.2004.09.001.

Kmiecik, M. and Soares, C.G. (2002), "Response surface approach to the probability distribution of the strength of compressed plates", *Marine Struct.*, **15**(2), 139-156. https://doi.org/10.1016/S0951-8339(01)00024-7.

Koza, J.R. (1992), *Genetic programming: On the Programming of Computers by Means of Natural Selection*, The MIT Press, MA, USA.

Li, H., Lü, Z. and Yue, Z. (2006), "Support vector machine for structural reliability analysis", *Appl. Math. Mech.*, **27**(10), 1295-1303. https://doi.org/10.1007/s10483-006-1001-z.

Li, L. (2012), *Sequential Design of Experiments to Estimate a Probability of Failure*, Supélec, France.

Liu, H., Ong, Y.S. and Cai, J. (2017), "A survey of adaptive sampling for global metamodeling in support of simulation-based complex engineering design", *Struct. Multidisciplinary Optimization*, 1-24. https://doi.org/10.1007/s00158-017-1739-8.

Liu, P.L. and Der Kiureghian, A. (1991), "Optimization algorithms for structural reliability", *Struct. Safety*, **9**(3), 161-177.

Liu, X., Wu, Y., Wang, B., Ding, J. and Jie, H. (2017), "An adaptive local range sampling method for reliability-based design optimization using support vector machine and Kriging model", *Struct. Multidisciplinary Optimization*, **55**(6), 2285-2304. https://doi.org/10.1007/s00158-016-1641-9.

Matheron, G. (1973), "The intrinsic random functions and their applications", *Adv. Appl. Probability*, 439-468. https://doi.org/10.2307/1425829.

McCulloch, W.S. and Pitts, W. (1943), "A logical calculus of the ideas immanent in nervous activity", *Bullet. Math. Biophys.*, **5**(4), 115-133. https://doi.org/10.1007/BF02478259.

Metropolis, N., Rosenbluth, A.W., Ronsenbluth, M.N., Teller, A.H. and Teller, E. (1953), "Equations of state calculations by fast computing machines", *J. Chem. Phys.*, **21**, 1087-1092. https://doi.org/10.1063/1.1699114.

Pan, Q. and Dias, D. (2017), "An efficient reliability method combining adaptive support vector machine and Monte Carlo simulation", *Struct. Safety*, **67**, 85-95. https://doi.org/10.1016/j.strusafe.2017.04.006.

Parsons, R. and Canfield, S. (2002), "Developing genetic programming techniques for the design of compliant mechanisms", *Struct. Multidisciplinary Optimization*, **24**(1), 78-86. https://doi.org/10.1007/s00158-002-0216-0.

Pradlwarter, H., Schueller, G., Koutsourelakis, P. and Charmpis, D. (2007), "Application of line sampling simulation method to reliability benchmark problems", *Struct. Safety*, **29**(3), 208-221. https://doi.org/10.1016/j.strusafe.2006.07.009.

Rackwitz, R. and Flessler, B. (1978), "Structural reliability under combined random load sequences", *Comput. Struct.*, **9**(5), 489-494. https://doi.org/10.1016/0045-7949(78)90046-9.

Rahman, S. and Wei, D. (2006), "A univariate approximation at most probable point for higher-order reliability analysis", *J. Solid Struct.*, **43**(9), 2820-2839. https://doi.org/10.1016/j.ijsolstr.2005.05.053.

Robert, C. and Casella, G. (2004), *Monte Carlo Statistical Methods*, *Vol. 319*, Springer, Germany.

Sacks, J., W. J. Welch, Mitchell, T.J. and Wynn., H.P. (1989), "Design and analysis of computer experiments", *Statistical Sci.*, **4**(4), 409-435.

Schueremans, L. and Van Gemert, D. (2005), "Benefit of splines and neural networks in simulation based structural reliability analysis", *Struct. Safety*, **27**(3), 246-261. https://doi.org/10.1016/j.strusafe.2004.11.001.

Schueremans, L. and Van Gemert, D. (2005). "Use of Kriging as Meta-model in simulation procedures for structural reliability", *International Conference on Struct. Safety and Reliability*, Rome, Italy, June.

Searson, D.P., Leahy, D.E. and Willis, M.J. (2010). "GPTIPS: an open source genetic programming toolbox for multigene symbolic regression", *Proceedings of the International Multiconference of Engineers and Computer Scientists*, Hong Kong, March.

Searson, D.P., Willis, M.J. and Montague, G. (2007), "Co-evolution of non-linear PLS model components", *J. Chemometrics*, **21**(12), 592-603. https://doi.org/10.1002/cem.1084.

Shanmugam, K. and Balaban, P. (1980), "A modified Monte-Carlo simulation technique for the evaluation of error rate in digital communication systems", *IEEE Transactions on Communications*, **28**(11), 1916-1924. https://doi.org/10.1109/TCOM.1980.1094613.

Song, H., Choi, K.K., Lee, I., Zhao, L. and Lamb, D. (2013), "Adaptive virtual support vector machine for reliability analysis of high-dimensional problems", *Struct. Multidisciplinary Optimization*, **47**(4), 479-491. https://doi.org/10.1007/s00158-012-0857-6.

Tan, X.-h., Bi, W.-h., Hou, X.-l. and Wang, W. (2011), "Reliability analysis using radial basis function networks and support vector machines", *Comput. Geotech.*, **38**(2), 178-186. https://doi.org/10.1016/j.compgeo.2010.11.002.

Teixeira, A.P. and Soares, C.G. (2010), *Response Surface Reliability Analysis of Steel Plates with Random Fields of Corrosion*, Taylor & Francis Group, London, United Kingdom.

Tvedt, L. (1990), "Distribution of Quadratic Forms in Normal Space — Application to Structural Reliability", *J. Eng. Mech.*, **116**(6), 1183-1197. https://doi.org/10.1061/(ASCE)0733-9399(1990)116:6(1183).

Valdebenito, M., Jensen, H., Hernández, H. and Mehrez, L. (2018), "Sensitivity estimation of failure probability applying line sampling", *Raliability Eng. Syst. Safety*, **171**, 99-111. https://doi.org/10.1016/j.ress.2017.11.010.

Waarts, P. (2000), "Structural reliability using finite element methods", *An appraisal of DARS: Directional Adaptive*, **515**.

Xiang, H., Li, Y., Liao, H. and Li, C. (2017), "An adaptive surrogate model based on support vector regression and its application to the optimization of railway wind barriers", *Struct. Multidisciplinary Optimization*, **55**(2), 701-713. https://doi.org/10.1007/s00158-016-1528-9.

Yeun, Y., Kim, B., Yang, Y. and Ruy, W. (2005), "Polynomial genetic programming for response surface modeling part 2: adaptive approximate models with probabilistic optimization problems", *Struct. Multidisciplinary Optimization*, **29**(1), 35-49. https://doi.org/10.1007/s00158-004-0461-5.

Yuan, X., Lu, Z., Zhou, C. and Yue, Z. (2013), "A novel adaptive importance sampling algorithm based on Markov chain and low-discrepancy sequence", *Aerospace Sci. Technol.*, **29**(1), 253-261. https://doi.org/10.1016/j.ast.2013.03.008.

Zhao, H., Ru, Z., Chang, X., Yin, S. and Li, S. (2014), "Reliability analysis of tunnel using least square support vector machine", *Tunnelling Underground Space Technol.*, **41**, 14-23. https://doi.org/10.1016/j.tust.2013.11.004.

Žiha, K. (1995), "Descriptive sampling in Structural Safety", *Struct. Safety*, **17**(1), 33-41. https://doi.org/10.1016/0167-4730(94)00038-R.

CC