IJBT (2018) 01: 12-22
RESEARCH ARTICLE

Decision-making under uncertainty

Monika[1] · Hao Quan[2] · Dipti Srinivasan[1]

Abstract Choosing actions on the basis of imperfect observations with unknown outcomes is called decision-making under uncertainty, and exists in many important problems. It unifies researchers across various disciplines to develop and model tools and methodologies to solve real-world decision-making problems under uncertainty. This paper proposes an efficient uncertainty modelling method using neural network-based prediction intervals (NN-based PIs) based on [1]. PIs are excellent tools to model uncertainties. Particle swarm optimisation (PSO)-based lower upper bound estimation (LUBE) method is used to construct NN-based PIs. Thereafter, a scenario generation method is used to generate scenarios from a list of PIs. These scenarios are further incorporated into the stochastic model for decision-making.

Keywords: Decision-making · Uncertainty · Neural networks · Particle swarm optimization · Lower Upper Bound Estimation · Prediction

1 Introduction

Every decision-making process holds a level of uncertainty. Uncertainty can never be eliminated, but can undeniably be condensed by using various methodologies and following some ideas like reducing time horizons for decisions, determining the worst case scenario, managing decisions adaptively, clarifying the uncertainty, increasing knowledge, and so on [2]. Uncertainty can be defined as the scenario of not having complete knowledge due to intrinsic deficiencies in acquired knowledge [3]. On the basis of its sources, we can classify uncertainty as ambiguity, approximations and likelihood. The ambiguity comes from the prospect of having multiple outcomes for processes or systems. The process of approximation can involve the use of imprecise semantics in language, approximate reasoning, and dealing with complexity, by emphasising relevance. Approximations can be viewed to include imprecision, abrasiveness and simplification. The likelihood can be defined in the context of chance, odds and gambling. Likelihood possesses the primary components of randomness and sampling [4]. Uncertainty can also be employed for characterizing the state of a system such as uncertainty of the result. The sources of uncertainty, while using in the field of modelling and computation of various aspects of the world

Monika:
sermo@nus.edu.sg

Hao Quan:
quanh@epgc.a-star.edu.sg

✉ Dipti Srinivasan:
dipti@nus.edu.sg

[1] Department of Electrical and Computer Engineering, National University of Singapore, Singapore
2 Experimental Power Grid Centre, Agency for Science, Technology and Research (A*STAR), Singapore

may include parameter uncertainty, model inadequacy, parametric variability, observation error, numerical uncertainty, interpolation uncertainty, etc. [1].

Decision-making under uncertainty (DMUU) modelling faces numerous challenges like what is uncertainty, level and extent of information available, approaches for analysing uncertainty, how to incorporate the uncertainty modelling methods into practices for decision-making and henceforth. There are two types of problems in quantification of uncertainty, forward uncertainty propagation and inverse uncertainty quantification. The first one focuses on the response of system for outputs when inputs are uncertain. The later one attempts to estimate bias correction and parameter calibration. The forward uncertainty propagation methods include probabilistic methodologies [5][6][7][8][9][10][11][12][13] such as simulation-based methods (Monte-Carlo simulations, importance sampling, etc.), local expansion-based methods (such as Taylor series, perturbation method, etc.), functional expansion-based methods (like Neumann expansion, polynomial chaos expansion, etc.), most probable point models, numerical integration based models; and non-probabilistic approaches [14] like interval analysis, fuzzy theory, possibility theory and evidence theory.

The essential categorization of probability is the probability distribution function (PDF), which can be defined as the relative likelihood for an arbitrary variable to take on a given value. The PDF is non-negative ubiquitously and the integral of PDF over the entire space is equal to one. The density of a random variable, X is f, where f is a non-negative Lebesgue-integrable function, if [15]:

$$P\,[A \leq X \leq B] = \int_A^B f(x)dx$$

Let us say that F is the cumulative distribution function (CDF) of X, then F can be represented as:

$$F(x) = \int_{A-\infty}^{x} f(u)du$$

If f is continuous at x then,

$$f(x) = \frac{d}{dx} F(x)$$

Instinctively, one can think that $f(x)dx$ is probability of X falling within the infinitesimal interval $[x, x + dx]$. Figure 1 shows the uniform, Gaussian and binomial PDF and CDF.

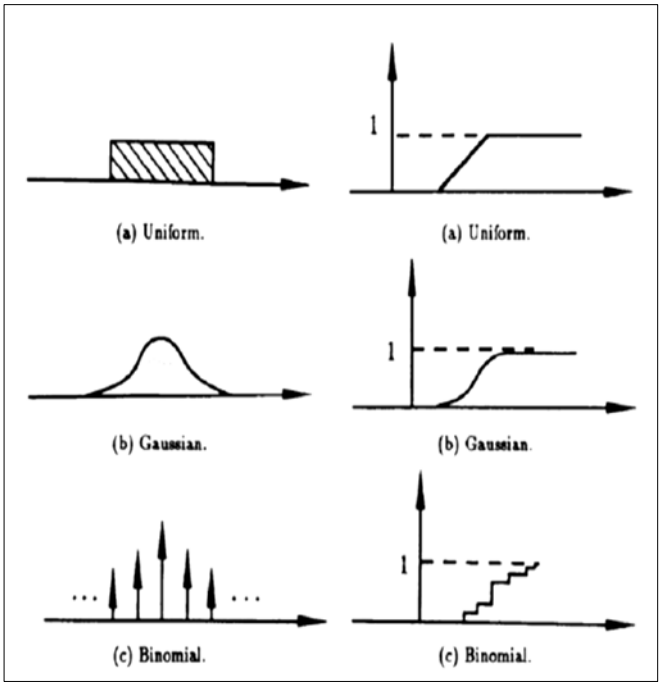

Fig. 1 The uniform, Gaussian and binomial PDF (left) and CDF (right)

Let us say that X is a random variable whose value is given and its distribution states a probability density function *f*, then the expected value of *X* (if it exists) can be calculated as,

$$E\,[X] = \int_{-\infty}^{\infty} x\, f(x) dx$$

In regression applications of neural networks, point forecasts have been mainly used. Nevertheless, owing to certain disadvantages of point forecasts like decrease in reliability of point forecasts with increase in level of certainty and lack in information about prediction accuracy, prediction intervals (PIs) are becoming increasingly popular.

The rest of the paper is organized as follows. PIs and their evaluation indices are introduced in Section 2. PSO-based LUBE method to construct PIs and stochastic modelling for decision-making are introduced in Sections 3 and 4 respectively. Finally, Section 5 draws the conclusion.

2 Prediction intervals

Prediction interval is a terminology used in statistics, which infers to an estimate of interval for future observations, based on the available observations. A PI consists of lower and upper bounds that brace a future unknown target value with a certain probability ((1-α) %) which is known as the confidence level. A typical PI comprises of three units: the upper bound, lower bound and the coverage probability. With the use of high quality PIs, it becomes convenient for the decision makers to confidently draw up future plans, better manage risks, and maximize their benefits. Like point forecasts are evaluated using the indices MSE (Mean Square Error) and MAPE (Mean Absolute Percentage Errors), on the similar grounds PIs are evaluated using the indices PICP (Prediction Interval Coverage Probability), PINAW (prediction interval normalized average width) and PINRW (prediction interval normalized root-mean-square width) whose mathematical representations are given below:

$$PICP = \frac{1}{n} \sum_{i=1}^{n} c_i$$

$$PINAW = \frac{1}{nR} \sum_{i=1}^{n} (U_i - L_i)$$

$$PINRW = \frac{1}{R} \sqrt{\frac{1}{n} \sum_{i=1}^{n} (U_i - L_i)\text{^}2}$$

$$CWC = PINAW + \phi(PICP) e^{-\eta(PICP - \mu)}$$

where n is the number of samples, U_i and L_i are the upper and lower bounds of PIs, c_i is an indicator of the PI coverage, R is the range of the underlying targets. PINAW and PINRW correspond to the 1-norm and 2-norm of the width of PIs. The arrangement of PINAW is analogous to MAPE when point forecasting is concerned, and it gives equal weights to all widths of PIs. The other index PINRW is functionally analogous to MSE, and amplifies wider intervals. For higher quality of prediction intervals, larger PICP and narrower PINRW form the essential properties. These properties of PICP and PINRW are contradictory, as demanding a larger PICSP will always result in wider PINRW, and tapering PINRW may lead to an unsatisfactorily low PICP. When considered from the optimization point of view, this is a two-objective problem. For the simplification and comprehensive comparisons for different PIs, it is necessary to transform this primary multi-objective problem into a single-objective one. An example of cost function for complete assessment of PIs is coverage width-based criterion (CWC). In CWC, a step-function, $\Phi(PICP)$, is defined whose value is majorly dependent on the fulfilment of PICP, with the parameters μ and η, controlling the position and amount of jump in CWC.

3 PSO-based LUBE method

In this paper, a method called LUBE (Lower Upper Bound Estimation) is proposed for the construction of prediction intervals (PIs). This method is preferred over the traditional methods for reasons such as:

- It is simpler and paradigms PIs with higher quality in a single step. On the contrary, traditional methods first perform point forecasting followed by construction of PIs.
- As a matter of fact, the real-world data with uncertainty such as wind power, solar energy, market response, etc. is unstable and intermittent in nature. Thus, assumptions about distribution of data seems problematic and doubtful. The LUBE method is a nonparametric method and hence no assumption on data distribution is made. Whereas, traditional methods constantly consider a parametric distribution (e.g., Normal) of data and then attempt to find its parameters for construction of intervals.
- LUBE method has significantly lower computational burden for PI construction when compared to alternative methods, which owes to the fact that the developed NN directly generates PIs. Alternatively, other approaches first need to linearize NN models or calculate complex matrices such as the Jacobian and Hessian matrices.

PIs with a high coverage probability and narrow width are expected for decision-making. However, these two aspects of PIs contradict with each other. For example, increasing the coverage probability will also widen the PIs while squeezing the PIs may lead to a lower coverage probability. This multi-objective problem can be transformed into a constrained single-objective problem by reassigning the coverage probability and then treating it as a hard constraint and the only objective is to minimize the width of PIs.

The basic concept of LUBE method is to adopt a NN with two outputs to directly generate the upper and lower bounds of PIs. The first and second outputs correspond to the upper and lower bounds of PIs separately. The symbolic NN with two outputs for the LUBE method is shown in Figure 2.

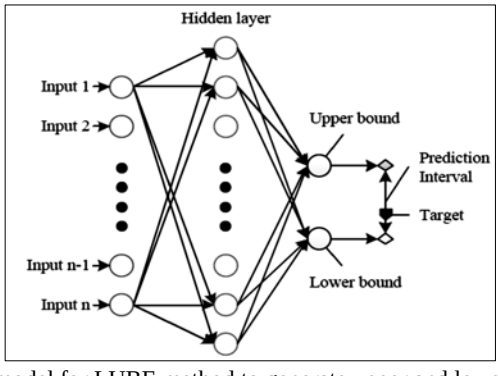

Fig. 2 NN model for LUBE method to generate upper and lower bounds of PIs

The plotting between the inputs and outputs of the system is shown below [16]:

$$y_i = f_1\left(\sum_{j=1}^{N_h}\left(w_{ij} f_2\left(\sum_{k=1}^{N_i} v_{jk} x_k + b_{vj}\right) + b_{wi}\right)\right)$$

where y_i is the output of the i^{th} node on the output layer; x_k is the input of the k^{th} node in the input layer; w_{ij} represents the connection weight between nodes in the hidden and output layers; v_{jk} is the connection weight between nodes in the input and hidden layers; and b_{wi} and b_{vj} are bias terms that represent the threshold of the transfer function f_1 and f_2. The number of nodes in the input, hidden and output layers are N_i, N_h and N_o respectively. In LUBE method, N_o = 2, and y_1, y_2 correspond to the upper and lower bounds of PIs. The LUBE method generates the upper and lower bounds directly, making PI construction as simple as point forecasts.

Neural Networks make the quality of PIs sensitive to their own structure. The question is how to determine layers and number of neurons in each layer. Too small or too large NNs have a low simplification power and often suffer under-fitting or over-fitting problems. The determination of the NN structure is key to the successful construction of high quality PIs. In this paper, fully connected feed-forward three-layered NNs are chosen and the number of neurons in the hidden layer is varied from 1 to 20. In order to determine an optimal structure of NNs, a k-fold cross validation method is employed. This method relies on a k-fold cross correlation. In order to maintain independence of the training and test, the k-fold cross validation is implemented on the training set. The entire training set is divided equally into k complementary folds, out of which, usually k-1 are used for training the candidate NNs, and the remaining fold is used for the purpose of validation [17]. For avoiding biased sampling, every structure of NN is trained and validated for k times using k different sub-training and validation datasets. A 5-fold cross validation is implemented here. Traditionally error-based measures such as MAPE and MSE are used for selecting the optimal structure of NNs. The focus of this paper is on construction of high quality PIs. Therefore, it is more reasonable to determine the optimal structure of NNs directly using PI-based evaluation indices, such as PICP, PINAW and CWC. Determination of the optimal NN structure needs to balance between the network complexity, generalization and learning capacity of NNs.

Meta-heuristic approaches are computational intelligence methods inspired by natural evolution or social behaviour. These methods include expert systems [18], fuzzy logic [19], GA [20][21], evolutionary programming, simulated annealing, tabu search, PSO [22], ant colony optimization and differential evolution [23]. PSO (Particle Swarm Optimization) is a parameter optimization technique which is employed to solve the problem of optimizing the PIs.

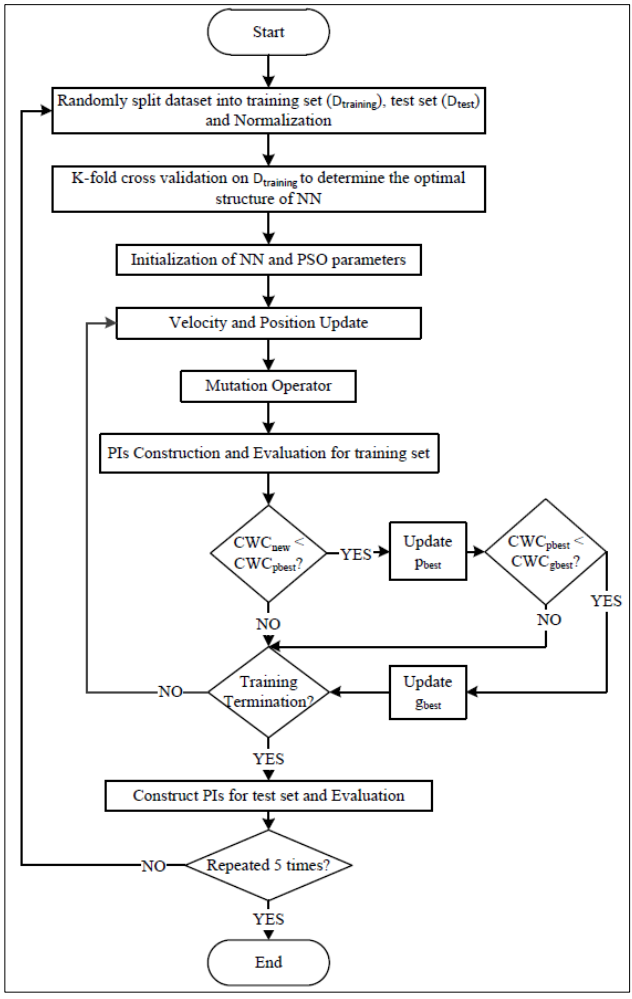

Fig. 3 PSO-based LUBE method for construction and evaluation of PIs

The objective of using PSO has mainly two aspects. Firstly, PSO is used to solve the single-objective problem which is to handle the constraints and optimize the objective. Secondly, PSO with mutation operator is used as the training algorithm through optimizing the connection weights [24] of NN models.

The flow chart of the proposed PSO-based LUBE method is shown in Figure 3.

The data is first split into data set and test set randomly, whereafter the training dataset is normalised, following the same procedure for test dataset. The training set ($D_{training}$) is further split into sub-training sets and validation sets using the 5-fold cross validation method. Median value of CWC is used to determine the optimal structure of NNs. Then the parameters of NNs and PSO are initialised. A bad initialisation may not converge to a very good result, hence initialisation of parameters is very important for this algorithm. After investigation [25], it was found that Nguyen-Widrow (NW) method obtains the best and most stable results. Hence NW method is used in this algorithm for NN weight initialisation. PSO parameter initialization consists of

particle position and velocity initialization. Since the weights of NN connection are represented as the position of particles, hence position initialization gets completed in weight initialization. Zero symmetric numbers are used to initialize the particle velocity which is a process of random initialization. Velocity and position update are the core of the PSO algorithm. The particles will exchange their "*findings*" with each other in the update process. In this way, the information gets exchanged efficiently throughout the whole swarm. The classic formulas for velocity and position update [26] are shown below:

$$v_n\,(t+1) = W.\,v_n\,(t) + C_1\,rand()\,(p_{best,n} - x_n(t)) + C_2\,rand()\,(g_{best,n} - x_n(t))$$
$$x_n(t+1) = x_n(t) + v_n(t+1)$$

where v_n is the particle velocity in the n^{th} dimension, *rand()* is a random number between 0 and 1, W is a scaling factor, and C_1 and C_2 are scaling factors that determine the relative "*pull*" of p_{best} and g_{best}.

Besides the two updates, the ranges for velocity and position are limited to V_{max} and X_{max} separately. Selection, crossover and mutation form the three main operators in GA. Mutation operator, which helps achieve diversity in GA, is integrated into PSO. This integration strongly enhances the searching capacity and avoids being trapped into local optima. In flow chart shown in Figure 3, Gaussian mutation is added to each connection weight after the position update. The mean and standard deviation of Gaussian distribution are the weight value and 10% of that weight value respectively. The mutation rate exponentially decreases as the optimization continues. In the validation step, the training and validation sets are combined together to train the NN. LUBE method is applied to construct new PIs after updating the NN connection weights. PI assessment indices (PICP and PINRW) are calculated. p_{best} is the personal best value of each particle and g_{best} denotes the best value of the whole swarm. These values are updated during the process of training. Once the training process terminates, the g_{best} value is chosen to generate PIs for the test set. PICP and PINAW instead of PINRW are calculated and recorded. For the comprehensive evaluation purpose, CWC is also calculated. The process is repeated a couple of times. This PSO-based LUBE method is convergent.

4 Stochastic modeling for decision-making

In case of point forecasts, there is only one forecast value which can be used directly for decision-making. But in case of PIs, even a single level PI consists of three components namely upper bound, lower bound and corresponding confidence level. As decision-making becomes difficult in the presence of two bounds, linkage between the PIs and decision-making becomes important. In this paper, a computational framework is proposed for building up this important linkage by introducing a scenario generation method and stochastic modelling.

In a decision-making process, for a given lead time, a single level PI does not suffice making an optimal decision. Theoretically, a sufficient number of PIs or quantiles are required to represent any type of probabilistic distributions. Using the PSO-based LUBE method the various levels of PIs are constructed, which are applied to estimate the empirical cumulative distribution function (ECDF) of unknown

probabilistic distributions of the parameters under consideration. Now the challenge is to apply these PIs to mathematical models for decision-making. As multilevel PIs are presented and PIs suffer from multi-valued problem for decision-making, a scenario generation technique is proposed to properly represent the uncertainties and these scenarios are mathematically involved into the stochastic model for computational purpose for decision-making. The steps involved in implementation of the proposed generation scenario include:

1. Apply the PSO-based LUBE method to make forecasts for a list of PIs with multiple confidence levels;
2. If the constructed PIs are assumed to be central PIs, then each PI can be uniquely decomposed into two quantiles. The *(1 - α) %* PIs consists of two bounds, i.e. the lower and upper bounds. They correspond to the *(α/2) %* and *(1- α/2) %* quantiles respectively;
3. Obtain discrete points on ECDF and generate the ECDF curve fitting;
4. Monte Carlo simulation is applied to generate scenarios from the fitted ECDF.

This method can be easily implemented, avoiding the complex computation like the covariance matrix. It also avoids strong assumptions on probabilistic distributions, the only assumption being that the constructed PIs are central PIs. This method builds an important bridge between the PIs and the scenarios used in stochastic model. These generated scenarios are further computationally involved for decision-making.

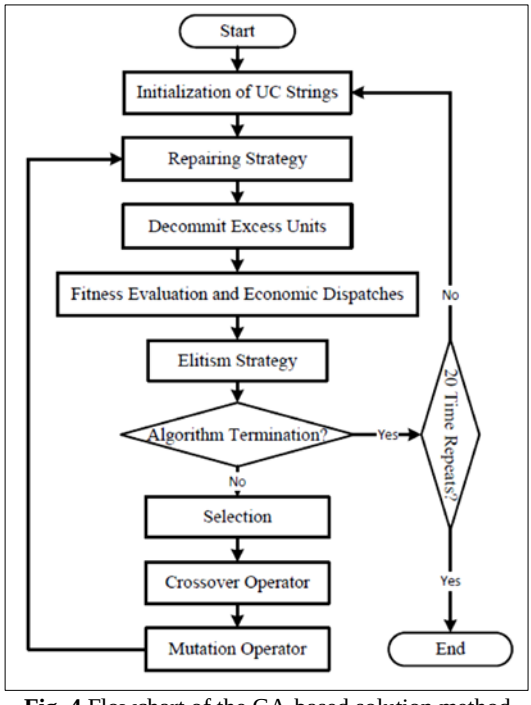

Fig. 4 Flowchart of the GA-based solution method

To solve the stochastic problem, a GA-based solution is proposed whose flowchart is presented in Figure 4, which shows an example of this GA-based solution method applied to unit commitment problems in electrical power systems. But this framework can be fitted to other optimization problems as well.

Initialization is crucial to the repeatability and success rate of the GA-based solution method for the stochastic problem. During initialization, all the constraints are not considered, hence the chromosomes need repairing periodically. The fitness function is defined in the following equation and is positively oriented, i.e., the greater, the better. *A* is a system-dependent constant used to avoid getting too small fitness values. *E(X, P)* is the objective of the stochastic problem representing expected cost.

$$Fitness = \frac{A}{E\ (X,P)}$$

Elitism is a useful strategy frequently used in GA. The idea is to reserve the best one or more populations in previous generation and directly copy them to the next generation without any modifications. In this way, the best fitness values will not decrease with increasing number of iterations. In this paper, the two fittest solutions are reserved first, and the two worst solutions after the selection are replaced by the two elitism members. *q*-tournament selection is implemented in the GA. The most used value for q is 2. The larger value the q is, the higher the selection pressure becomes in the population. A hefty value of q means the entire population is dominated by the members with high fitness values. There is a need to maintain a balance between the selection pressure and the diversity of the population.

The crossover operator is a two-point crossover. Two-point crossover calls for two points to be selected on the parent chromosome strings, under a certain crossover probability. Everything between the two points is swapped amongst the parent binary strings, rendering two child chromosomes.

Adaptive mutation is chosen here, whose mutation rate exponentially decreases with the increase of the number of generations. For each mutation, a random number is generated. If the random number is smaller than the mutation rate, the value on this bit is flipped; otherwise, the value remains the same as before.

The termination criterion can be set as the reach of the maximum number of iterations or few improvements are made in a certain number of generations.

5 Conclusion

In this paper, PIs are proposed to quantify uncertainties from forecasting. We propose a new method called PSO-based LUBE method for creating multilevel PIs, which involves neural networks. These PIs are then used for generation of scenarios using the ECDF and Monte Carlo Simulation. These generated scenarios are thereafter employed into a stochastic model for decision-making.

References

1. H. Quan: Uncertainty modeling in distributed power systems. Ph. D diss. National University of Singapore (2014)
2. Making effective decisions in high uncertainty, http://www.decision-making-solutions.com/decision-making-in-uncertainty.html
3. B. M. Ayyub, G. J. Klir: Uncertainty modelling and analysis in engineering and the sciences. Chapman & Hall/CRC Press (2006)
4. N. Attoh-Okine, B. M. Ayyub: Applied research in uncertainty modeling and analysis, ser. International Series in Intelligent Technologies. Springer (2005)
5. S. H. Lee, W. Chen: A comparative study of uncertainty propagation methods for black-box type functions. ASME 2007 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference. American Society of Mechanical Engineers. 1275–1284 (2007)
6. S. Raychaudhuri: Introduction to Monte Carlo simulation. Simulation Conference, WSC 2008. Winter. IEEE. 91–100 (2008)
7. P. J. Smith, M. Shafi, H. Gao: Quick simulation: A review of importance sampling techniques in communications systems. IEEE Journal on Selected Areas in Communications. **15**(4), 597–613 (1997)
8. C. G. Bucher: Adaptive sampling an iterative fast Monte Carlo procedure. Structural Safety. **5**(2), 119–126 (1988)
9. B. Lallemand, G. Plessis, T. Tison, and P. Level: Neumann expansion for fuzzy finite element analysis. Engineering computations. **16**(5), 572–583 (1999)
10. K. Sepahvand, S. Marburg, and H.-J. Hardtke: Uncertainty quantification in stochastic systems using polynomial chaos expansion. International Journal of Applied Mechanics. **2**(2), 305–353 (2010)
11. X. Du and W. Chen: A most probable point-based method for efficient uncertainty analysis. Journal of Design and Manufacturing Automation. **4**(1), 47–66 (2001)
12. F. Xiong, K. Guo, and W. Zhou: Uncertainty propagation techniques in probabilistic design of multilevel systems. International Conference on Quality, Reliability, Risk, Maintenance, and Safety Engineering (ICQR2MSE). IEEE, 874–878 (2011)
13. H. Xu and S. Rahman: A generalized dimension-reduction method for multidimensional integration in stochastic mechanics. International Journal for Numerical Methods in Engineering. **61**(12), 1992–2019 (2004)
14. C. Chatfield: Calculating interval forecasts. Journal of Business and Economic Statistics. **11**(2), 121-135 (1993)
15. H. Tijms: Understanding probability. Cambridge University Press (2012)
16. C.-H. Chen, J.-C. Wu, and J.-H. Chen: Prediction of flutter derivatives by artificial neural networks. Journal of Wind Engineering and Industrial Aerodynamics. **96**(10-11), 1925 – 1937(2008)
17. J. Wright and M. Manic: Neural network architecture selection analysis with application to cryptography location. International Joint Conference on Neural Networks (IJCNN), 1–6 (2010)
18. P.-H.Chen: Two-level hierarchical approach to unit commitment using expert system and elite PSO. IEEE Transactions on Power Systems. **27**(2), 780–789 (2012)
19. B. Venkatesh, P. Yu, H. Gooi, and D. Choling: Fuzzy MILP unit commitment incorporating wind generators," IEEE Transactions on Power Systems, **23**(4), 1738–1746 (2008)
20. S. Kazarlis, A. Bakirtzis, and V. Petridis: A genetic algorithm solution to the unit commitment problem. IEEE Transactions on Power Systems. **11**(1), 83–92(1996)
21. K. Swarup and S. Yamashiro: Unit commitment solution methodology using genetic algorithm. IEEE Transactions on Power Systems. **17**(1), 87–91 (2002)
22. T. Logenthiran and D. Srinivasan: Particle swarm optimization for unit commitment problem. IEEE 11th International Conference on Probabilistic Methods Applied to Power Systems (PMAPS). 642–647 (2010)
23. X. Yuan, A. Su, H. Nie, Y. Yuan, and L. Wang: Application of enhanced discrete differential evolution approach to unit commitment problem. Energy Conversion and Management. **50**(9), 2449–2456 (2009)
24. Q. Sun, Y. Yu, Y. Luo, and X. Liu: Application of BFNN in power flow calculation in smart distribution grid. Neurocomputing. 125, 148–152 (2014)
25. A. Pavelka and A. Proch: Algorithms for initialization of neural network weights random numbers in matlab. Control Engineering. 2, 453–459(2004)
26. G. Pulido and C. Coello: A constraint-handling mechanism for particle swarm optimization. IEEE Congress on Evolutionary Computation, CEC 2004, 2, 1396–1403 (2004).