Single-loop reliability-based design optimization methods for single and multi-objective optimization

A thesis submitted
in partial fulfilment of the requirements
for the degree of
Doctor of Philosophy

by

RAKTIM BISWAS
(156103033)

to the

DEPARTMENT OF MECHANICAL ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY GUWAHATI
June 2022

Dedicated to

Maa, Baba, Amrita, and Dr. Deepak Sharma

CERTIFICATE

It is certified that the work contained in the thesis entitled "***Single-loop reliability-based design optimization methods for single and multi-objective optimization***", by "*Mr. Raktim Biswas*", has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Dr. Deepak Sharma

June, 2022. Department of Mechanical Engineering,

I.I.T. Guwahati

Acknowledgment

Foremost, I would like to express my sincere gratitude to my supervisor and mentor Dr. Deepak Sharma, for his support, encouragement, and guidance during my stay at Indian Institute of Technology, Guwahati. He has been a constant source of help and inspiration to me. He gave a lot of freedom in my course and research work, and has been extremely supportive and understanding at all times. I could not have imagined a better supervisor for my Ph.D. study.

Beside my supervisor, I would like to thank the members of my doctoral committee, Dr. Sashindra K. Kakoty, Dr. Sukhomay Pal, and Dr. Prakash Kotecha, for their valuable feedback, suggestions and questions that were crucial to the completion of this thesis.

My sincere thanks also goes to Dr. Arunasis Chakraborty and Dr. Prakash Kotecha for allowing me to attend their classes informally.

I thank my fellow lab mates in computational mechanics and optimization lab: Shashikant, Shubhojit, Utpal, Vishal and Sai for the stimulating discussion and all the fun we had in the last seven years. I would also like to thank Shiv and Aritra for their support and friendship that made my stay memorable and joyful.

Most importantly, I would like to thank my parents and Amrita, for everything they have done for me, without them none of this would have been possible.

Raktim Biswas
IIT Guwahati
June, 2022

Abstract

Real world engineering system involves uncertainty either due to lack of exact knowledge of the physical system or due to working environment. Therefore, the solution evolved by solving such system with deterministic design optimization leads to failure of design. Reliability-based design optimization (RBDO) method acts as an efficient tool for designing such systems. Among all the methods used for solving RBDO problems, single-loop method is one of the most efficient methods. However, these methods are prone to converge to a local solution and often faces an issue of oscillation while solving probabilistic performance functions. This thesis aims to address these issues by proposing efficient and accurate single-loop RBDO methods. In the first part of the thesis, a single-loop sequential optimization with approximate reliability analysis is proposed. To maintain the accuracy of the approximate reliability analysis, conjugate gradient search direction is used to approximate and update the most-probable target point (MPTP). Further, a shifting vector is used to shift the probabilistic performance function. In order to minimize the effect of oscillation while obtaining MPTP, an approximate single-loop chaos control method is proposed. An oscillation criterion is developed to track the oscillation of MPTP in every iteration. When MPTP starts oscillating, the chaos control theory is used. Otherwise, conjugate gradient search direction is used to update the MPTP.

The above mentioned methods use numerical optimization techniques that require gradient information to obtain the optimal solution. Therefore, there is always a chance that a solution may stuck to local optimal solution. In the second part of the thesis, a single-loop RBDO formulation is developed by using shifting vector approach for achieving feasibility for violated performance functions. The formulation incorporates target and trial vectors of differential evolution (DE) for guiding the algorithm. DE is further made adaptive by designing a heuristic parameter that controls two mutation operators for both exploration and exploitation of search space.

As most of the real-world problems consist of multiple objectives apart from uncertainty, in the third part of the thesis a single-loop formulation for multi-objective RBDO is proposed to obtain a reliable Pareto-optimal front. The proposed multi-objective RBDO formulation

is coupled with the chaos control theory so that oscillation during the convergence of MPTP can be reduced. A heuristic parameter is estimated in every generation using hypervolume performance indicator for generating mutant vectors from different variants of DE. Further, the concepts of shifting vector approach and chaos control theory are also introduced with multi-objective RBDO formulation.

All the proposed methods are tested using different types of mathematical and engineering RBDO examples from the literature. The reliability of all the obtained solutions by the proposed methods are verified using Monte Carlo simulation (MCS). Also, the proposed methods are compared with existing RBDO methods from the literature. The results demonstrate that single-loop-based proposed formulations and methods generate the reliable optimal solutions in less function evaluations.

Raktim Biswas
IIT Guwahati
June, 2022

Publication

Journal Publications

Published

- Raktim Biswas and Deepak Sharma (2021). A single-loop shifting vector method with conjugate gradient search for reliability-based design optimization. *Engineering Optimization* 53(6), 1044-1063, DOI: 10.1080/0305215X.2020.1770745.
- Raktim Biswas and Deepak Sharma (2023). An approximate single-loop chaos control method for reliability-based design optimization using conjugate gradient search directions. *Engineering Optimization* 55:3, 382-398, DOI: 10.1080/0305215X.2021.2007242.
- Raktim Biswas and Deepak Sharma. A single-loop reliability-based design optimization using adaptive differential evolution. *Applied Soft Computing* 132, 109907 (Jan 2023). DOI: doi.org/10.1016/j.asoc.2022.109907
- Raktim Biswas and Deepak Sharma. (2023) Single-loop multi-objective reliability-based design optimization using chaos control theory and shifting vector with differential evolution. *Mathematical and Computational Applications* 28(1):26, DOI: doi.org/10.3390/mca 28010026

Under review

- Raktim Biswas and Deepak Sharma. Chaos control assisted single-loop multi-objective reliability-based design optimization using differential evolution. *Swarm and Evolutionary Computation*, Manuscript number - SWEVO-D-21-00793R2

Conference Publication

- Raktim Biswas and Deepak Sharma. A Single-Loop Reliability-based Design Optimization Method using Iteratively updating Hessian. *EUROGEN, 19*

Contents

List of Figures

List of Tables

Acronym

Acronym

RIA	Reliability Index Approach
PMA	Performance Measure Approach
MPP	Most Probable Point
MPTP	Most Probable Target Point
SLShV-CG	Single-loop Shifting Vector with Conjugate Gradient
ASLCC	Adaptive Single-loop Chaos Control
AMV	Advanced Mean Value
ASORA	Approximate Sequential Optimization and Reliability Assessment
CC	Chaos Control
SLA	Single-loop Approach
HCC	Hybrid Chaos Control
SORA	Sequential Optimization and Reliability Assessment
SLADE	Single-loop Adaptive Differential Evolution
SLMDE	Single-loop Multi-objective Differential Evolution
RBDO	Reliability-based Design Optimization
HV	Hypervolume
PDF	Probability Density Function
FORM	First-order Reliability Method
SORM	First-order Reliability Method
DE	Differential Evolution
KKT	Karush-Kuhn Tucker
HLRF	Hasofer Lind Rackwitz Fiessler
BFGS	Broyden-Fletcher-Goldarb-Shanno
CMV	Conjugate Mean Value
CGA	Conjugate Gradient Analysis
RMV	Relaxed Mean Value
AAMV	Adjusted Advance Mean Value

HSMV	Hybrid Self Adjusted Mean Value
MCC	Modified Chaos Control
SMCC	Self Adaptive Modified Chaos Control
SR-1	Symmetric Rank One
PPM	Probabilistic Performance Measure
SAP	Sequential Approximate Programming
HSAP	Hybrid Sequential Approximate Programming
ADA	Adaptive Decoupling Approach
MORRAP	Multi-objective reliability-redundancy allocation problem
OSV	Optimal Shifting Vector
ISV	Incremental Shifting Vector
SLSV	Single-loop Single Vector
SDLM	Single-loop Deterministic Method
RDS	Reliable Design Space
GA	Genetic Algorithms
PSO	Particle Swarm Optimization
ABC	Artificial Bee Colony
MPGA	Multi-Population Genetic Algorithm
IRSM	Improved Support Vector Regression
WSM	Weighed Simulation Method

Chapter 1

Introduction

An optimization formulation of a design problem consists of minimization or maximization of objective function that is subjected to constraints. This type of formulation is known as deterministic formulation as randomness is not considered in the design variables or parameters. The solution evolved by solving this optimization problem is often located on the constraint boundary. However, many engineering design problems involve uncertainty in design variables due to which the optimal solution of deterministic optimization problem becomes infeasible. Such uncertainty can occur due to manufacturing imperfections, and inaccuracy in geometric dimension, material properties, modelling, etc. Therefore, it is necessary to design the optimization problem that can be robust and reliable.

The well known "safety factors" and "empirical relations" can be used with deterministic optimization to design under uncertainty. These traditional design procedures generate a conservative solution and do not implement uncertainties directly. Therefore, the solution is calibrated to match the reliability of solution, but for extreme cases catastrophic failure (Choi et al., 2006) may occur. Moreover, when scattering of values, i.e., standard deviation about the mean is considered, the safety factors or empirical approach would fail or may produce too conservative design. To overcome this issue, scientists and engineers have applied probabilistic and statistical methods with optimization.

Probabilistic engineering design is a mathematical design methodology used for producing high quality design products, which are vulnerable to uncertainty and also minimize or reduce the effects of uncertainty. It can be broadly categorised into three parts, namely:

- Reliability-based design: It seeks that the probability of failure should be less than some acceptable small value. It focuses on high reliability and low risk of design solution.

- Robust design: It improves the quality of the design by minimizing the effect of uncertainty without eliminating the cause and improves the robustness of design performance.

- Design for six Sigma (DFSS): It is generally used for product development, which links business and consumer needs to critical product function, detailed design, tests, attributes and verification. All three methodologies share a common goal of improving product or process performance, but they differ in their specific focus. DFSS is primarily concerned with reducing defects and improving quality, while reliability-based design focuses on improving the reliability of the product or system. Robust design, on the other hand, aims to minimize the impact of variability on product performance.

Obtaining the optimal solution for reliability-based design and robust design leads to the field of reliability-based design optimization (RBDO) and robust design optimization (RDO), respectively. The uncertainty considered in these models is characterized using probability distribution, which can be obtained by statistical models. The details are given in the following section.

1.1 Uncertainty and reliability analysis

1.1.1 Uncertainty

Uncertainties can be involved in the problem during designing stage, manufacturing process, or during service of the component. In design stage, it is considered by mathematical modelling of the system. During manufacturing, it is reflected by tolerance, which arises due to lack of advanced technologies for manufacturing process. The boundary conditions and environmental effects like vibration, temperature changes, etc., are the major factors leading to service uncertainties. They can effect the performance of the design as well as reduce lifetime.

One of the major concerns is how to quantify uncertainties in design optimization? Over the last few decades, many theories have been proposed by researchers to introduce uncertainties in designing process. In general, these uncertainties can be categorised as aleatory uncertainty and epistemic uncertainty. Aleatory uncertainty refers to inherent variability or randomness in the system or process being modeled. It is sometimes called “irreducible” uncertainty because it is a fundamental property of the system that cannot be eliminated through better understanding or more data. Epistemic uncertainty, on the other hand, refers to uncertainty arising from lack of knowledge or understanding of the system being modeled. It is sometimes called “reducible”

uncertainty because it can be reduced by improving our understanding of the system or by obtaining more data. These uncertainties must be quantified and categorised before design optimization.

The uncertainty analysis depends mainly on the probabilistic theories that enables us to understand the impact on the design solution. Probability theory can represent uncertainties as random variables that is denoted by an uppercase (e.g., $\mathbf{X}$). The nature of randomness and the information on probability are represented by the probability density function (PDF). The PDF represents the relative frequency of certain realizations for random variables and the information also introduces the use of random variables, processes, and fields to represent uncertainty. The center of the PDF indicates the most probable point (MPP) and the extreme ends of the PDF show less probable events. MPP has the highest probability density on the performance function in the standard normal space. In this thesis, we estimate MPP efficiently and effectively for highly non-linear functions and consider only aleatory type of uncertainty, i.e., distribution type is available.

1.1.2 Reliability

Reliability is the probability that a system will perform its function over a specified period of time and under specified service conditions. The goal is to predict the probability of failure for a given system. An engineering system depends on many uncertain factors, such as loads, boundary conditions, stiffness, etc. The design solution should consider these uncertainties and meet certain requirements within acceptable value of certainty in terms of constraints satisfaction. Reliability is estimated by the prediction and calculation of the probability of performance function violations at any stage during a system's life. The probability of the occurrence of an event such as 'performance function violation' is a numerical measure of probability of failure of the system. After probability of failure (P_f) is determined, the reliability is estimated by $(1 - P_f)$ and design alternatives are chosen to improve reliability and minimize the risk of failure.

1.2 Reliability-based design optimization

Reliability-based design optimization (Tu et al., 1999) deals with obtaining optimal reliable solution that has a low failure probability. The uncertainties are considered by replacing the deterministic constraints with probabilistic constraints. These constraints are often termed as

performance function or limit state function in the literature. The RBDO formulation is given in equation (1.1).

$$\begin{aligned} \text{Min.} \quad & f(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{X}}), \\ \text{s.t.} \quad & P[G_i(\mathbf{d}, \mathbf{X}) \leq 0] \leq P_{f_i}^t = \Phi(-\beta_i^t), \quad i = 1, \ldots, I, \\ & h_j(\mathbf{d}) \leq 0, \qquad\qquad\qquad\qquad\quad j = 1, \ldots, J, \\ & \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{L})} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{U})}, \\ & \mathbf{d}^{(\mathrm{L})} \leq \mathbf{d} \leq \mathbf{d}^{(\mathrm{U})}, \end{aligned} \tag{1.1}$$

where $f(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{X}})$ is the objective function, $G_i(\mathbf{d}, \mathbf{X})$ is the $i-$th performance/constraint function that depends on the vector of deterministic design variables $\mathbf{d}$, and the vector of random variables $\mathbf{X} \in \mathrm{R}^N$. $\boldsymbol{\mu}_{\mathbf{X}}$ is the mean value of $\mathbf{X}$. $h_j(\mathbf{d})$ is the $j-$th deterministic constraints, and, L and U in the superscript represent the lower and upper limits of the vectors. $\Phi(\cdot)$ represents the standard normal cumulative distribution function, β_i^t is the target reliability index of $i-$th performance function, $P[\cdot]$ is the probability operator that represents the probability of performance function $(G_i(\mathbf{d}, \mathbf{X}) \leq 0)$. $P[\cdot]$ should be less than the target failure probability, $P_{f_i}^t$ and is estimated by multidimensional integral as shown in equation (1.2).

$$P[G_i(\mathbf{d}, \mathbf{X}) \leq 0] = \int \cdots \int_{G_i(\mathbf{d}, \mathbf{X}) \leq 0} f_{\mathbf{X}}(\mathbf{X}) \mathrm{d}\mathbf{x}, \tag{1.2}$$

where $f_{\mathbf{X}}(\mathbf{X})$ is the joint probability density function of $\mathbf{X}$. The multidimensional integral in equation (1.2) is equal to the cumulative distribution function "$F_{G_i}(0)$" such that $F_{G_i}(0) = \Phi(-\beta_i^t)$. An exact evaluation of equation (1.2) is difficult to obtain and, therefore, different methods have been adopted to approximate this integral. First-order reliability method (FORM) (Hasofer and Lind, 1974, Rackwitz and Flessler, 1978) and second-order reliability method (SORM) (Kiureghian et al., 1987) are commonly used to approximate this integral and perform reliability analysis. FORM often provides inadequate accuracy but is widely used in RBDO applications due to its computational efficiency. On the other hand, the accuracy of SORM is better than FORM with an expense of extra computational cost.

FORM and SORM are based on a nested optimization structure commonly known as double-loop method. The outer loop optimizes the objective function, and the inner loop performs reliability analysis for each probabilistic performance function, thereby making the double-loop methods computationally expensive. In addition, the computational efficiency also decreases exponentially as the number of random variables and performance function increases. The computational efficiency has been improved by scientists and researchers in which the re-

liability analysis and optimization loop are decoupled and performed sequentially. Further, an approximate deterministic performance function is established to replace the probabilistic performance function and to eliminate the reliability analysis loop. This method is known as single-loop method. However, it has convergence issue for non-linear performance function that results in oscillation around the solution (Yang and Yi, 2009).

1.3 Motivation

RBDO is an optimization problem that aims to minimize or maximize an objective function subject to probabilistic constraints on the system's performance. Reliability analysis is a crucial step in RBDO, but it is computationally expensive, particularly for nonlinear performance functions. To improve the computational efficiency of RBDO methods, researchers have proposed using approximate reliability analysis to obtain the most probable point (MPP). However, while this approach can reduce computational costs, it can also lead to convergence issues. One key knowledge gap in the current state of RBDO research is to develop more accurate and efficient methods for reliability analysis, especially for complex and nonlinear systems.

This thesis aims to address this challenge by developing a novel single-loop method by incorporating conjugate gradient search direction to approximate MPP. The shifting vector approach is adopted with the proposed single-loop method that shifts the violated performance function toward the feasible direction. The combination of conjugate gradient search direction and shifting vector approach makes the proposed single-loop method better and computationally efficient. When the proposed method has been tested on various RBDO examples, it is found that the obtained RBDO solution gets oscillated for highly non-linear functions. This lead to the second objective of this thesis in which an oscillation criterion is proposed to trace oscillation among MPPs. In every iteration, MPP is estimated using conjugate gradient search direction. When oscillation among MPPs is observed in the current and previous iterations, chaos control theory is used to update the current MPP.

The above proposed single-loop methods have significantly improved the performance in generating reliable solutions with less function evaluation. However, they face other challenge while dealing with non-linear objective and performance functions. For example, the conjugate search directions is susceptible to converge to the local optimal solution. This challenge can be addressed by using the global optimizer, such as evolutionary algorithm. From the literature, it is found that most of the studies are done on coupling evolutionary optimization

techniques with double-loop methods to obtain global reliable solution. Since both evolutionary optimization techniques and reliability analysis are computationally expensive, it is needed to develop efficient RBDO method. This leads to the third objective of the thesis, in which a global optimization technique called differential evolution (DE), which is a population based meta-heuristic optimization algorithm, is modified and used for obtaining the reliable solution. DE is made adaptive by proposing a heuristic parameter that can help DE to perform different mutation operators. The method is also coupled with shifting vector approach to move the violated performance functions toward the feasible direction.

As most of the real-world problems consist of multiple objectives, the next objective is to develop a multi-objective RBDO formulation in which the reliability analysis is approximated using Karush Kuhn Tucker (KKT) optimality conditions. The concept of chaos control theory is used with the formulation for estimating the new most probable target point in order to reduce its oscillation during convergence. The method is developed using multi-objective DE algorithm that is designed using non-dominated sorting and crowding distance. The algorithm is also made adaptive by introducing a heuristic parameter for generating mutant vectors from different variants of DE using hypervolume performance indicator. Further, another new formulation for multi-objective RBDO is proposed using shifting vector approach and adaptive multi-objective DE for generating the reliable Pareto-optimal solutions.

1.4 Objectives of the thesis

The aim of this thesis is to develop single-loop RBDO method that cogenerate reliable and optimal solution and should also be computationally efficient. Following are the objectives of the thesis.

1. Develop a single-loop method using conjugate gradient search and shifting vector approach for improving the accuracy of reliable solution.

2. Enhance the proposed single-loop method using chaos control theory for estimating MPP efficiently for non-linear performance functions.

3. Develop a hybrid single-loop RBDO method incorporating the concept of KKT optimality conditions and shifting vector approach with adaptive DE for global optimization.

4. Develop a multi-objective RBDO formulation using chaos control theory and solve it using DE with adaptive mutation scheme.

5. Develop a hybrid multi-objective RBDO formulation incorporating modified chaos control theory and shifting vector approach, and solve it using adaptive DE.

1.5 Organization of the thesis

The thesis is organized in the following chapters.

- **Chapter 2** An exhaustive literature review of RBDO methods and different types of reliability analyses are presented in this chapter.
- **Chapter 3** The formulations of the single-loop RBDO using conjugate gradient direction, shifting vector, and chaos control theory are described in detail. The oscillation criterion for MPP is also discussed. The results and discussion are presented on various RBDO examples.
- **Chapter 4** The formulation of hybrid single-loop RBDO is discussed. The adaptive DE and its implementation with different mutation schemes are discussed in detail. The results are presented and compared with the existing RBDO methods.
- **Chapter 5** The detailed discussion of the single-loop multi-objective RBDO is presented. Efficient reliability analysis with modified chaos control theory and shifting vector is discussed. The implementation of DE is discussed by incorporating hypervolume performance indicator. The obtained Pareto-optimal solutions are generated and compared with the double-loop multi-objective DE with PMA for reliability analysis.
- **Chapter 6** The thesis is concluded with a note on future work in this chapter.

Chapter 2

Literature Review

An exhaustive literature review of RBDO methods is presented in this chapter. It starts with brief introduction of single objective deterministic optimization and RBDO. Thereafter, various RBDO methods are discussed along with basic types of reliability analyses in order to have a complete understanding of the subject. Finally, different types of meta-heuristic algorithms used to solve single and multi-objective RBDO are discussed.

2.1 Single-objective reliability-based design optimization

2.1.1 Deterministic design optimization

A deterministic design optimization formulation can be written as

$$\begin{aligned} &\text{Min. } f(\mathbf{d}), \\ &\text{s.t.: } G_i(\mathbf{d}) \leq 0, \qquad i = 1, \ldots, I, \\ &\mathbf{d}^{(\mathrm{L})} \leq \mathbf{d} \leq \mathbf{d}^{(\mathrm{U})}, \end{aligned} \tag{2.1}$$

where the deterministic design variable vector $\mathbf{d} = [d_1, d_2, \ldots, d_N]^T$ is determined to solve the optimization problem. $f(\mathbf{d})$ is the objective function, $G_i(\mathbf{d})$ is the $i-$th inequality constraint, $\mathbf{d} = [d_1, d_2, \ldots, d_N]^{\mathrm{T}}$ is the N-dimensional design vector, and $\mathbf{d}^{(\mathrm{L})}$ and $\mathbf{d}^{(\mathrm{U})}$ are the lower and upper limits of the design vector $\mathbf{d}$, respectively. According to equation (2.1) if $G_i \leq 0$ at a given solution, the constraint has satisfied the condition. Generally, numerical optimization algorithms are used for obtaining optimal solution satisfying these constraints. Apart from that, meta-heuristic algorithms can also be used for obtaining global optimal solution.

2.2 Reliability-based design optimization (RBDO)

In RBDO, the deterministic variables become random variables and the deterministic constraints become probabilistic constraints that are often termed as probabilistic performance function. The formulation of RBDO (Tu et al., 1999, Youn et al., 2003) is given in equation (1.1). As discussed in section 1.2, an exact evaluation of equation (1.2) is challenging and therefore, $G_i(\mathbf{X})$ is approximated by using the first-order or second-order Taylor series. Subsequently, probabilistic performance function can be computed by using reliability analysis methods like reliability-index approach (RIA) or performance measure approach (PMA).

Performing the reliability analysis effectively and efficiently is the main challenge of RBDO as it requires many function evaluations. Also, the accuracy and stability of RBDO problem (Aoues and Chateauneuf, 2010) depend on the execution of the reliability analysis. Many approaches have been developed in the past that can be divided into three categories:

- Double-loop method
- Decoupled-loop method
- Single-loop method

In the following section these methods are briefly discussed and relevant literature is presented.

2.2.1 Double-loop RBDO methods

Double-loop method involving nested loops to solve RBDO problems is shown in Figure 2.1. The optimization loop has an inner loop that performs reliability analysis of the current solution either using RIA or PMA. The outer loop estimates the design inputs that minimize the objective function and satisfy the probabilistic constraint. Two of the most commonly used reliability analysis methods are discussed in the following sections.

2.2.2 First-order reliability methods (FORM)

In FORM (Hasofer and Lind, 1974, Rackwitz and Flessler, 1978, Tu et al., 1999) the constraint $G(\mathbf{X})$ is approximated by first-order Taylor series expansion after transforming normal random variables in X-space to the standard normal random variable in U-space (Madsen et al., 1986). If all random variables are mutually independent, i.e, the correlation coefficient is zero, the

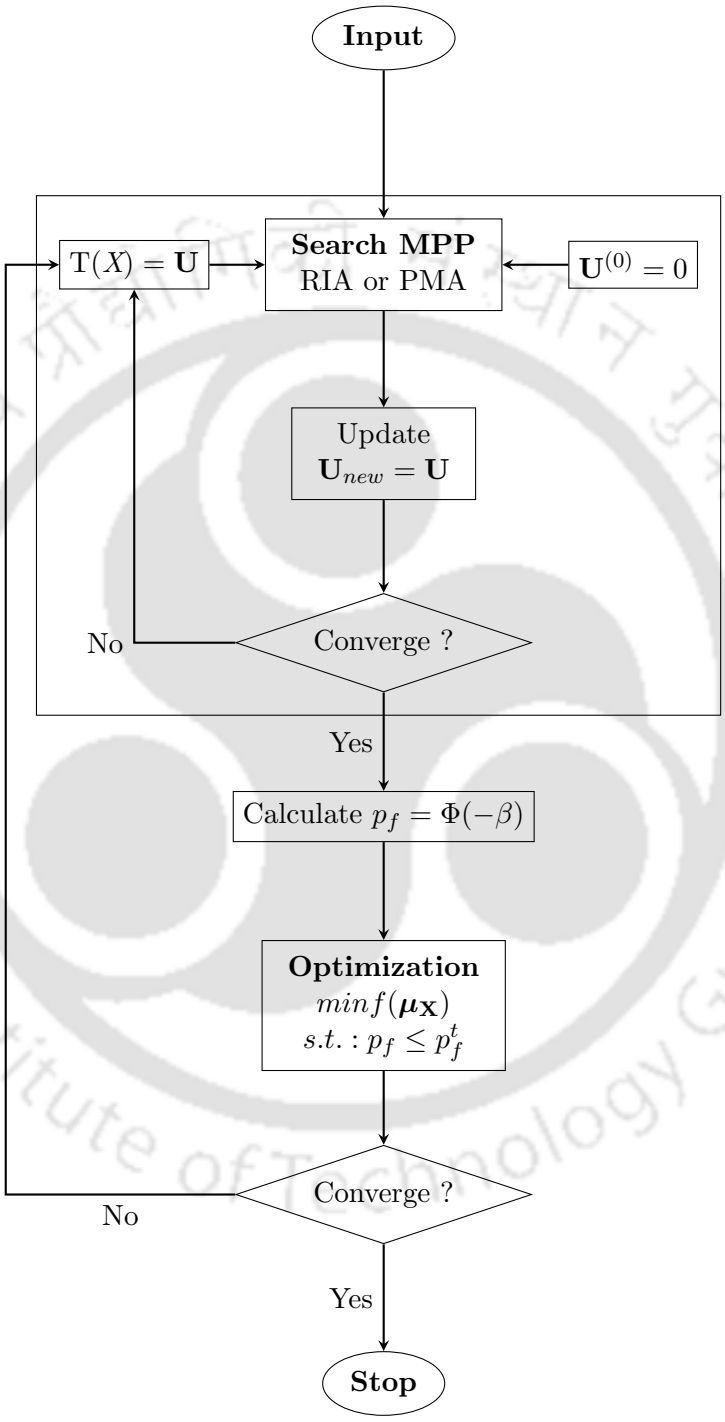


Figure 2.1: Flowchart of double-loop method

transformations can be simplified as

$$\begin{aligned} U &= \Phi^{-1}(F_X(X)), \\ X &= F_X^{-1}(\Phi(U)). \end{aligned} \tag{2.2}$$

After transformation the performance function $G(\mathbf{X})$ can be represented in the U-space as $G(\mathbf{U})$. The point on the hypersurface ($G(\mathbf{U}) = 0$) with the maximum joint probability density is known as most probable point (MPP). The MPP is located on $G(\mathbf{U}) = 0$ with the minimum distance from the origin in the U-space. This process of obtaining MPP is known as reliability-index approach (RIA). On the other hand, the optimum value evolved by minimizing $G(\mathbf{U})$ subjected to $\|\mathbf{U}\| = \beta^t$ also gives MPP. This problem is the inverse of RIA and is solved by performance measure approach (PMA). Brief discussion on both RIA and PMA are presented in the following section.

2.2.2.1 Reliability index approach (RIA)

The probabilistic performance function of equation (1.1) can be transformed into reliability index as given in equation (2.3).

$$\hat{\beta}_i = -\Phi^{-1}(F_{G_i}(0)) \geq \beta_i^t, \qquad i = 1, \ldots, I, \tag{2.3}$$

where $\hat{\beta}_i$ and β_i^t are the reliability index and target reliability index for $i-$th performance function, respectively. These indexes are used to replace the probabilistic performance function of equation (1.1) by equation (2.3) and thus, the RBDO formulation is modified as

$$\begin{aligned} &\text{Min. } f(\boldsymbol{\mu}_{\mathbf{x}}), \\ &\text{s.t. } \hat{\beta}_i \geq \beta_i^t, \qquad\qquad i = 1, \ldots, I, \\ &\boldsymbol{\mu}_{\mathbf{x}}^{(\mathrm{L})} \leq \boldsymbol{\mu}_{\mathbf{x}} \leq \boldsymbol{\mu}_{\mathbf{x}}^{(\mathrm{U})}. \end{aligned} \tag{2.4}$$

RIA utilises first-order Taylor series expansion to calculate the reliability index, $\hat{\beta}_i$. For this calculation, a random variable vector ($\mathbf{X}$) is transformed to the standard normal random variable vector ($\mathbf{U}$), which can be performed by Rosenblatt transformation (Rosenblatt, 1952) or Nataf transformation (Liu and Kiureghian, 1986)[$\mathbf{U} = \mathrm{T}(\mathbf{X})$ or $\mathbf{X} = \mathrm{T}^{-1}(\mathbf{U})$]. Transformations of some common type of distributions are given in Table 2.1 and is obtained by transforming original design space to the standard normal design space, which is shown in Figure 2.2. The reliability index $\hat{\beta}_i$ is evaluated by solving an optimization problem (Rackwitz

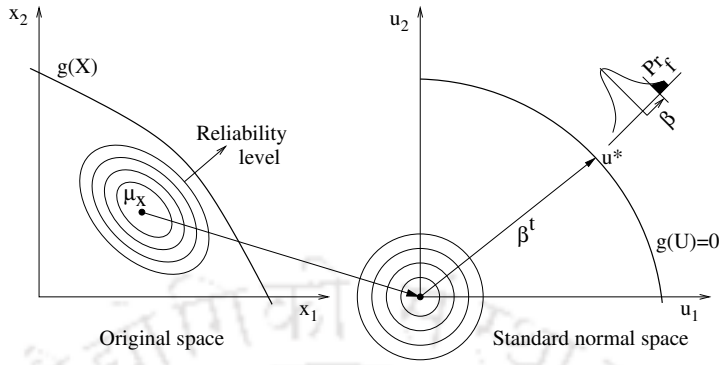


Figure 2.2: Transformation from the original design variable space to the standard normal space

Table 2.1: Transformation of U-space to X-space

Sl.No.	Distribution type	Transformation
1	Normal (μ, σ)	$\mu + \sigma \mathrm{U}$
2	Lognormal (μ, σ)	$e^{\mu+\sigma \mathrm{U}}$
3	Uniform (a, b)	$a + (b-a)(0.5 + 0.5erf(\mathrm{U}\sqrt{2}))$
4	Weibull (λ, k)	$\lambda[-ln(\Phi(\text{-U})^a)]^{\frac{1}{k}}$
5	Gumbel (ν, α)	$\nu - \frac{1}{\alpha} ln[-ln(\Phi(\mathrm{U}))]$
6	Gumbel (a, b)	$ab(\mathrm{U}\frac{1}{\sqrt{9a}} + 1 - \frac{1}{9a})^2$
7	Gamma (a, b)	$ab(\mathrm{U}\sqrt{\frac{1}{9a}} + 1 - \frac{1}{9a})^3$
8	Exponential (λ)	$-\frac{1}{\lambda} log(0.5 + 0.5erf(\mathrm{U}/\sqrt{2}))$

$erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt$

and Flessler, 1978), which is given in equation (2.5).

$$\begin{aligned} &\text{find } \mathbf{U}^* \\ &\text{min. } \|\mathbf{U}\| = \beta_i^t, \\ &\text{s.t.: } G_i(\mathbf{U}) = 0. \end{aligned} \tag{2.5}$$

The obtained optimum point $\mathbf{U}^*$ is called as the most probable failure point (MPFP) in the standard normal space. The flowchart of RIA is shown in Figure 2.3. The value of reliability index is updated in every iteration until the target reliability is achieved by the iterative points for the each performance function.

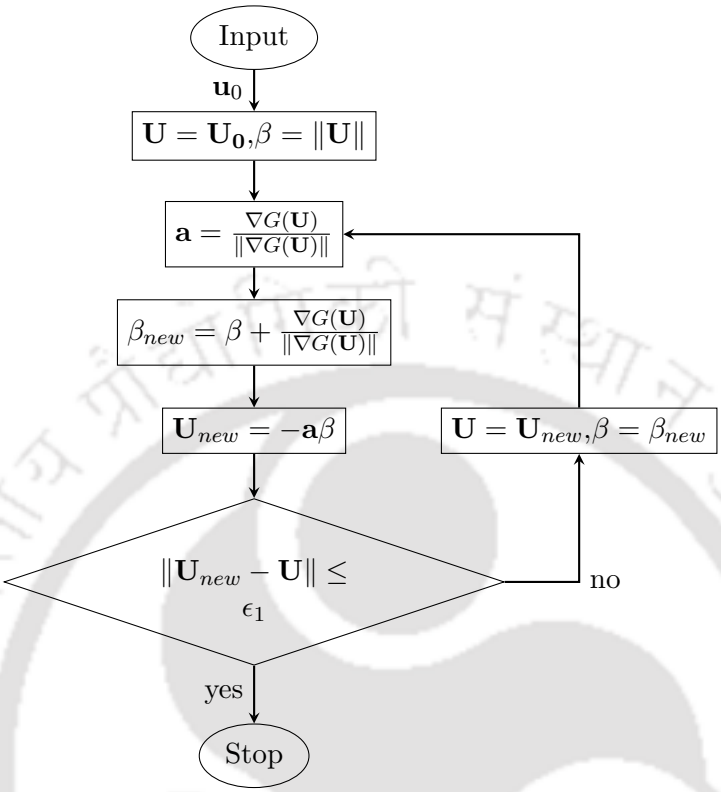


Figure 2.3: Flowchart of RIA

2.2.2.2 Performance measure approach (PMA)

Performance measure approach is an inverse analysis of RIA and is also known as inverse reliability analysis. This method is more efficient than RIA for the evaluation of $\mathbf{U}^*$, as the target reliability index is directly used in the formulation to obtain the optimal solution. The advantage of PMA is that nonlinear performance function is minimized subjected to simple constraint, which is easier to evaluate. Also, the direction vector is only to be determined by taking into consideration of hyperspherical equality constraint $\|\mathbf{U}\| = \beta_i^t$. Accordingly, the probabilistic performance function of RBDO formulation given in equation (1.1) can be transformed using equation (2.6).

$$G_i^p = F_{G_i}^{-1}(\Phi(-\beta_i^t)) \geq 0, \quad i = 1, \ldots, I, \tag{2.6}$$

where G_i^p is the $i-$th performance measure function. Therefore, the RBDO method based on PMA can be formulated as

$$\begin{aligned} &\text{Min. } f(\boldsymbol{\mu}_{\mathbf{x}}), \\ &\text{s.t. } G_i^p \geq 0, \qquad\qquad i = 1, \ldots, I, \\ &\quad \boldsymbol{\mu}_{\mathbf{x}}^{(\mathrm{L})} \leq \boldsymbol{\mu}_{\mathbf{x}} \leq \boldsymbol{\mu}_{\mathbf{x}}^{(\mathrm{U})}. \end{aligned} \tag{2.7}$$

The value of performance measure function can be calculated by solving another optimization problem given in equation (2.8).

$$\begin{aligned} &\text{find } \mathbf{U}^* \\ &\text{min. } G_i(\mathbf{U}), \\ &\text{s.t.: } \|\mathbf{U}\| = \beta_i^t, \quad i = 1, \ldots, I, \end{aligned} \tag{2.8}$$

where the obtained optimum point $\mathbf{U}^*$ is known as the most probable target point (MPTP) with target reliability β_i^t. The optimum value ($G_i(\mathbf{U}^*)$) is used as performance measure function G_i^p in equation (2.7) and is calculated as,

$$G_i^p = G_i(\mathbf{U}^*) = G_i(\mathbf{X}(\mathbf{U}^*)). \tag{2.9}$$

By applying Lagrangian multiplier method in equation (2.8), the iterative point (Wu et al., 1990) can be calculated using equation (2.10).

$$\mathbf{U}^{(k+1)} = -\beta_i^t \frac{\nabla G(\mathbf{U}^{(k)})}{\|\nabla G(\mathbf{U}^{(k)})\|}, \tag{2.10}$$

where ∇ is the gradient of function with respect to $\mathbf{U}$, and $\mathbf{U}^{(k)}$ is the MPTP at $k-$th iteration. The flowchart for PMA is shown in Figure 2.4. It can be seen that, based on a given value of β^t, $\mathbf{U}_{new}$ is updated subjected to the gradient of the performance function, and is terminated upon satisfying the termination criterion. Ramu et al. (2006) provide an overview of the various inverse measures used in the assessment of safety and describe their benefits when compared to direct measures such as probability of failure and reliability index.

2.2.3 Other FORM based methods

RIA-based FORM was proposed by Nikolaidis and Burdisso (1988) for solving RBDO problems with double-loop method. It estimates MPFP for each performance function in the inner loop

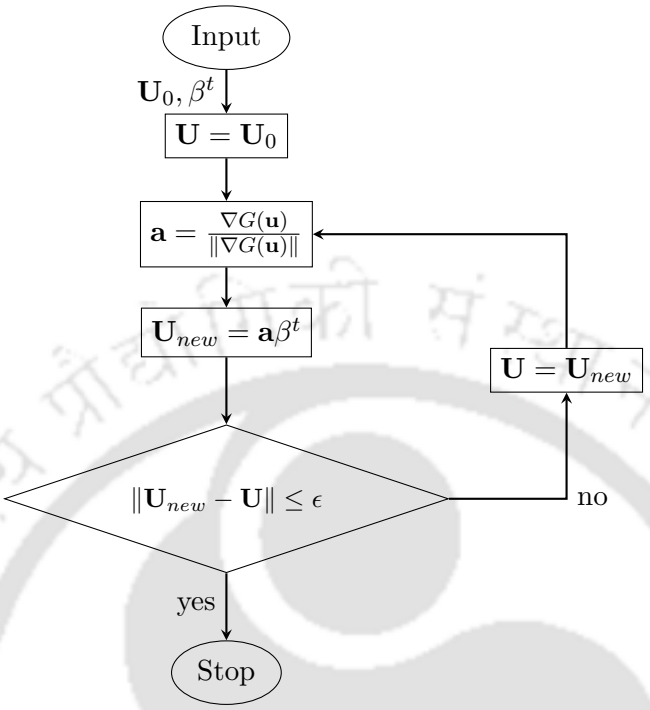


Figure 2.4: Flowchart for PMA

and uses the concept of FORM to evaluate probability of failure for obtaining mean values of random variables in the outer loop. Here, MPFP can also be evaluated by simulation based methods. A simplified safety index based approach (Reddy et al., 1994) was developed using an advanced second moment method to perform reliability analysis. The index values were interpolated using the information at the mean and MPFP. A linearized RIA was developed by Weiji and Li (1994) to approximate MPFP using linear programming tool for estimating optimal solution. An adaptive nonlinear approximation (Grandhi and Wang, 1998) constructed by the function values and first-order gradient information at two points are used to replace the performance function for conducting reliability analysis. Hasofer Lind Rackwitz Fiessler (HLRF) method was developed by Hasofer and Lind (1974) for solving reliability analysis as it converges quickly in most situations. When the performance function is highly non-linear, it shows slow convergence and may even diverge from the solution. HLRF algorithm in inverse-FORM using Broyden-Fletcher-Goldarb-Shanno (HLRF-BFGS) method developed by Periaro et al. (2015) considers the BFGS update formula to approximate the Hessian for estimating the MPFP. This method demonstrated better convergence than to HLRF.

An advanced mean-based method (AMV) was proposed by Wu et al. (1990), which estab-

lishes the full probability distribution to provide all the information required for reliability-based design. This method is more computationally efficient than traditional FORM. The iterative update to get MPP using AMV is given in equation (2.11).

$$\begin{aligned} \mathbf{U}^{*}_{\mathrm{MPP}} &= \beta^{t}\mathbf{n}(0), \\ \mathbf{n}(0) &= -\frac{\nabla_U G(\mathbf{U})}{\|\nabla_U G(\mathbf{U})\|}, \end{aligned} \tag{2.11}$$

where $\mathbf{U}^{*}_{\mathrm{MPP}}$ is the MPP of the performance function, $\mathbf{n}(\cdot)$ represents the normalized direction of performance function, $G(\mathbf{U})$ is the performance function at iterative point $(\mathbf{U})$, ∇ is the gradient operator with respect to U and $\|\cdot\|$ represents the norm of a vector. Moreover, it updates the direction vector in each iteration only at the current MPP as

$$\begin{aligned} &\mathbf{U}^{(1)}_{\mathrm{AMV}} = \mathbf{U}^{*}_{\mathrm{MPP}}, \qquad && \mathbf{U}^{(k+1)}_{\mathrm{AMV}} = \beta^{t}\mathbf{U}^{(k)}_{\mathrm{AMV}}, \\ &\text{where} && \mathbf{U}^{(k)}_{\mathrm{AMV}} = -\frac{\nabla_U G(\mathbf{U}^{(k)}_{\mathrm{AMV}})}{\|\nabla_U G(\mathbf{u}^{(k)}_{\mathrm{AMV}})\|}. \end{aligned} \tag{2.12}$$

As AMV method uses current MPP, it demonstrates slow convergence and sometimes even oscillates due to high non-linearity in the performance function. This issue has been resolved by using conjugate mean value (CMV) method. CMV updates the current MPP by using both current and previous MPP information. The new search direction is obtained by summing normalised gradient of consecutive three iterations, i.e, $\mathbf{n}(\mathbf{u}^{(k-2)}_{CMV}), \mathbf{n}(\mathbf{u}^{(k-1)}_{CMV})$ and $\mathbf{n}(\mathbf{u}^{(k)}_{CMV})$ with an equal weight. It is given as

$$\mathbf{U}^{(k+1)}_{CMV} = \beta_t \frac{\mathbf{n}(\mathbf{U}^{(k-2)}_{CMV}) + \mathbf{n}(\mathbf{U}^{(k-1)}_{CMV}) + \mathbf{n}(\mathbf{U}^{(k)}_{CMV})}{\|\mathbf{n}(\mathbf{U}^{(k-2)}_{CMV}) + \mathbf{n}(\mathbf{U}^{(k-1)}_{CMV}) + \mathbf{n}(\mathbf{U}^{(k)}_{CMV})\|} \quad , \text{for } k \geq 2, \tag{2.13}$$

$$\mathbf{n}(\mathbf{u}^{(k)}_{CMV}) = -\frac{\nabla_U G(\mathbf{u}^{(k)}_{\mathrm{CMV}})}{\|\nabla_U G(\mathbf{u}^{(k)}_{\mathrm{CMV}})\|},$$

where $\mathbf{U}^{(k+1)}_{CMV}$ is the MPP obtained by CMV at $(k+1)-$th iteration. The stability of the CMV has been improved for concave function and is found to be inefficient for convex function. In hybrid-mean value (HMV) method (Youn et al., 2003), CMV is selectively used for concave functions. This is done by identifying the nature of performance function for identifying it as convex or concave type.

HMV is found sensitive to initial design point. To improve its stability, conjugate gradient analysis (CGA) method (Ezzati et al., 2015) was proposed. In this method, conjugate gradient direction is used to update the MPTP. The gradient of performance function is calculated at every iterative point until convergence criterion is satisfied.

Relaxed mean value (RMV) was proposed by Keshtegar and Lee (2016) to evaluate both convex and concave probabilistic constraints using PMA. A relaxed factor within the range of 0 to 2 is adaptively used in inequality criterion to improve the computational efficiency and accuracy. The relaxed factor is developed using steepest descent direction and based on the sufficient descent criterion for PMA based double-loop method that is given as

$$\begin{aligned}
\mathbf{U}_{RMV}^{(k+1)} &= \beta^t \frac{\tilde{\mathbf{U}}_{RMV}^{(k+1)}}{\|\tilde{\mathbf{U}}_{RMV}^{(k+1)}\|}, \\
\tilde{\mathbf{U}}_{RMV}^{(k+1)} &= (1-\alpha^{(k)})\mathbf{U}_{RMV}^{(k)} + \alpha^{(k)}\mathbf{U}_{AMV}^{(k)}, \\
\alpha_{max} &= \min(\alpha^{(1)}, \beta^t); \alpha^{(k)} = \frac{\alpha_{max}}{C^m},\ m = 1, 2, \ldots,
\end{aligned} \tag{2.14}$$

where C is adaptive coefficient between 1.005 and 1.11, α is the relaxed factor, m is the iterative progress within MPP search, $\mathbf{U}_{RMV}^{(k+1)}$ and $\tilde{\mathbf{U}}_{RMV}^{(k+1)}$ are point candidates for beta hypersphere based on normalized search direction and relaxed line search based on steepest descent direction, respectively. The sufficient descent condition (Shi et al., 2010) is as follows

$$\|\tilde{\mathbf{U}}_{RMV}^{(k+1)} - \mathbf{U}_{RMV}^{(k+1)}\| < \|\mathbf{U}_{RMV}^{(k)} - \mathbf{U}_{RMV}^{(k-1)}\|. \tag{2.15}$$

An adjusted advance mean value approach (AAMV) was proposed by Keshtegar and Hao (2016). It uses the concept of hybrid single and double-loop (HSD) to have similar computational efficiency of single-loop method and accuracy of the double-loop method. The modified direction with adjusted step size ($\lambda = \exp(-\delta k)$) is represented as

$$\begin{aligned}
\mathbf{U}_{AAMV}^{(k+1)} &= \beta^t \frac{\tilde{\mathbf{U}}_{A}^{(k+1)}}{\|\tilde{\mathbf{U}}_{A}^{(k+1)}\|}, \\
\tilde{\mathbf{U}}_{A}^{(k+1)} &= \mathbf{U}^{(k)} + \lambda[\beta^t \alpha^{(k)} - \mathbf{U}^{(k)}],
\end{aligned} \tag{2.16}$$

where $\mathbf{U}_{AAMV}^{(k+1)}$ is the iterative point to find MPTP using AAMV method, δ is an adjusted parameter [0.05,0.1], $\tilde{\mathbf{U}}_{A}^{(k+1)}$ is the modified search direction using adjusted step size, and $\alpha^{(k)}$ is unit vector for descent direction. If the descent direction holds true, $\mathbf{U}^{(k+1)} = \mathbf{U}_{AMV}^{(k+1)}$;

otherwise, $\mathbf{U}^{(k+1)} = \mathbf{U}^{(k+1)}_{AAMV}$. The criterion employed to select single-loop or double-loop method is given as

$$\begin{aligned} &|\mathbf{U}^{(k+1)} - \mathbf{U}^{(k)}\| < 0.1|\mathbf{U}^{(k)} - \mathbf{U}^{(k-1)}\|, \quad \text{if single-loop method is used,} \\ &|\mathbf{U}^{(k+1)} - \mathbf{U}^{(k)}\| \geq 0.1|\mathbf{U}^{(k)} - \mathbf{U}^{(k-1)}\|, \quad \text{if double-loop method is used.} \end{aligned} \tag{2.17}$$

If double-loop method is used, MPTP is calculated by AAMV method for highly nonlinear probabilistic performance function.

AMV is later improved with self adjusted step size that is known as self adjusted mean value method (SMV) (Keshtegar and Hao, 2017). It is combined with AMV based on sufficient descent condition to form hybrid self adjusted mean value (HSMV) method. The self adjusted step size is given by as

$$\lambda^{(k)} = C(\lambda^{(k-1)})^P, \tag{2.18}$$

where C is adjusted coefficient $[0.9, 1]$, and P is the power factor $[1.1, 1.5]$. Here, $\mathbf{U}_{AMV}$ is used if the sufficient condition is satisfied; otherwise, MPTP is estimated by using equation (2.18) in equation (2.16).

In most of the cases, MPTP estimated by AMV method for highly nonlinear performance function leads to periodic and chaotic solution. To address this issue, a chaos control (CC) method (Yang and Yi, 2009) was proposed by extending the stability transformation method. The MPTP is estimated in the U-space as follows

$$\begin{aligned} \mathbf{U}^{(k+1)}_{CC} &= \mathbf{U}^{(k)}_{CC} + \lambda C\left[f(\mathbf{U}^{(k)}) - \mathbf{U}^{(K)}_{CC}\right], \\ f(\mathbf{U}^{(k)}) &= \mathbf{U}^{(k+1)}_{AMV}, \end{aligned} \tag{2.19}$$

where $\mathbf{U}^{(k+1)}_{CC}$ is the iterative progress of point to estimate MPTP by chaos control method at $(k+1)-$th iteration, C is the involutory matrix, $f(\mathbf{U}^{(k)})$ is the vector of response function which is estimated by MPTP of AMV method, and λ is the chaos control factor within $[0, 1]$.

To improve the convergence rate of CC method, Meng et al. (2015) proposed modified chaos control (MCC) method that estimates MPTP by extending the search on $\beta-$hypersphere by

using the following formulation.

$$\begin{aligned}
&\mathbf{U}_{MCC}^{(k+1)} = \beta^t \frac{\tilde{\mathbf{n}}^{(k+1)}}{\|\tilde{\mathbf{n}}^{(k+1)}\|}, \\
&\hat{f}(\mathbf{U}^{(k)}) = \mathbf{U}_{AMV}^{(k+1)}, \\
&\tilde{\mathbf{n}}^{(k+1)} = \mathbf{U}_{MCC}^{(k)} + \lambda C \left[\hat{f}(\mathbf{U}^{(k)}) - \mathbf{U}_{MCC}^{(K)}\right],
\end{aligned} \tag{2.20}$$

where $\mathbf{U}_{MCC}^{(k+1)}$ is the iterative point for estimating MPTP by MCC method, $\tilde{\mathbf{n}}^{(k+1)}$ is descent direction that is updated using the expression of MPTP of CC method, and the vector of response function $\hat{f}(\mathbf{U}^{(k)})$ is mapped to $\mathbf{U}_{AMV}^{(k+1)}$. MCC was found inefficient in dealing with convex performance function. Hence, a hybrid chaos control (HCC) method was proposed in which the method adaptively selects MCC or AMV method depending on the concave or convex nature of the performance function, respectively.

Another method popularly known as adaptive chaos control (ACC) (Li et al., 2015) was proposed based on CC method. It has similar structure of HCC method but the difference lies in updating scheme of chaos control factor (λ). In adaptive hybrid approach (AHA), the double-loop or single-loop method is implemented heuristically and adaptively. The ACC is used to locate the MPTP for double-loop method and AMV is selected in case of single-loop method. The selection criterion is based on maximum number of iterative points generated in the infeasible region. If the iterative point converges gradually, single-loop method is used for solving RBDO problem.

The effectiveness of CC and MCC methods are found dependent on chaos control factor. These methods may fail during convergence if high value of control factor is selected. This issue was resolved by self adaptive modified chaos control (SMCC) method (Keshtegar et al., 2017). The chaos control factor is updated the points in every iteration to estimate MPTP that is obtained as

$$\begin{aligned}
&\mathbf{U}_{SMCC}^{(k+1)} = \beta^t \frac{\tilde{\mathbf{n}}_s^{(k+1)}}{\|\tilde{\mathbf{n}}_s^{(k+1)}\|}, \\
&\tilde{\mathbf{n}}_s^{(k+1)} = \mathbf{U}_{SMCC}^{(k)} + \lambda_s C \left[\hat{f}(\mathbf{U}_{(k)}) - \mathbf{U}_{SMCC}^{(K)}\right],
\end{aligned} \tag{2.21}$$

where $\mathbf{U}_{SMCC}^{(k+1)}$ is the iterative point or estimate MPTP by SMCC method, $\tilde{\mathbf{n}}_s^{(k+1)}$ is enhanced descent direction that is updated using self adaptive factor λ_s and is given in equation (2.22)

$$\lambda_s = \min(\lambda_{int}, \lambda_k) \tag{2.22}$$

$\lambda^{(0)}$ and $\lambda^{(k)}$ are the dynamical control factor and initial control factor, respectively. These factors are given as

$$\begin{aligned}\lambda^{(0)} &= \min\left\{\frac{1}{\delta\beta^{t^2}}\left|\left[1-\frac{\mathrm{G}(\mathbf{U}_1)}{\mathrm{G}(\mathbf{U}_0)}\right]\right|, 1\right\},\\ \lambda^{(k)} &= \frac{2}{\|D_k\|^2}\left|\left[1-\frac{\mathrm{G}(\mathbf{U}_1)}{\mathrm{G}(\mathbf{U}_0)}\right]\right|,\end{aligned} \tag{2.23}$$

where δ is the control factor that is estimated less than one, D_k is the line search vector that is computed using the AMV method, $\mathbf{U}_0 = \mu$ that is the mean values of random variable and $\mathbf{U}_1 = \mathbf{U}_{AMV}^{(k)}$.

A probabilistic sufficiency factor (Qu and Haftka, 2004) was proposed as a measure of safety level relative to a target safety level. This factor was obtained from the results of Monte Carlo simulation with minimal additional computation. It was demonstrated that the design response surface approximation could be more accurately fitted to the probabilistic sufficiency factor than the probability of failure or safety index. Furthermore, the probabilistic sufficiency factor provided additional information in low-probability regions. The probability of failure or safety index could not be estimated by Monte Carlo simulation with a given sample size. This information proved helpful in guiding the optimizer.

2.2.3.1 Second-order reliability methods

FORM linearises the higher-order nonlinear performance function to estimate the probability of failure and thus, increases the error in estimating MPP. For this reason, second-order reliability method (SORM) (Breitung, 1984, Tvedt, 1983) has been developed, which has better accuracy than FORM. SORM performs second-order Taylor series expansion to approximate the performance function and is represented in equation (2.24).

$$G(\mathbf{U}) \approx L(\mathbf{U}) = G(\mathbf{U}^*) + \nabla G(\mathbf{U}^*)(\mathbf{U}-\mathbf{U}^*)^T + \frac{1}{2}(\mathbf{U}-\mathbf{U}^*)\mathbf{H}(\mathbf{U}^*)(\mathbf{U}-\mathbf{U}^*), \tag{2.24}$$

where $\mathbf{H}(\mathbf{U}^*)$ is the Hessian matrix calculated at MPP $\mathbf{U}^*$. As SORM needs second order derivative to calculate Hessian, it requires more function evaluations than FORM (Lim et al., 2014, Meng et al., 2018).

In order to reduce the computational requirement of SORM (Lim et al., 2014), the Hessian matrix was updated by quasi-Newton approach. Instead of calculating the Hessian directly, a Symmetric rank one (SR-1) update was used that required computations similar to FORM, leading to an efficient and accurate reliability analysis. A generalized chi-squared distribution

is also used for better accuracy. In another study, the computational efficiency of SORM (Lim et al., 2016) was improved by approximating the Hessian matrix by reusing the derivatives and accumulating it from previous FORM based RBDO iteration. Therefore, additional function evaluation is not required. In this method also SR-1 update is used using the derivatives to approximate Hessian.

It can be observed that FORM and SORM perform complete reliability analysis in the inner loop of nested optimization procedure to obtain the reliable solution. As a result these RBDO methods becomes computationally inefficient. For improving the computational efficiency, reliability analysis loop is separated from the nested optimization and performed sequentially. These methods are termed as decoupled-loop methods that are discussed in the following section.

2.2.4 Decoupled-loop RBDO methods

Decoupled-loop methods aim to reduce the computational requirement of double-loop method by performing reliability analysis separately until the convergence is achieved. RBDO problem is converted into a series of deterministic optimization cycles, including the independent reliability analysis. Figure 2.5 shows the flowchart of a decoupled-loop method in which the reliability assessment and optimization are performed sequentially.

A sequential optimization and reliability assessment method (SORA) was proposed by Du and Chen (2004) in which a series of cycle performing optimization and reliability analysis are employed sequentially. In each cycle, optimization and reliability analysis loop are decoupled from each other. The reliability analysis can only be conducted after optimization. It also shifts the boundaries of the violated constraint towards the feasible direction based on the information from previous cycle by using a shifting vector.

An enhanced SORA (ESORA) was proposed by Huang et al. (2013) which considers constant and varying variances of random variables by maintaining the same framework of SORA. When the performance function is linear, RBDO is transformed into complete deterministic optimization problem. When the performance function is non linear, the gradient at the actual MPTP is approximated by the previous cycle values.

An approximate probabilistic performance measure (PPM) was proposed by Yi et al. (2016) to approximate the MPTP. In each cycle the MPTP is reserved to calculate the new MPTP of the next cycle. Therefore, evaluation of the performance function in the deterministic optimization is not required. Approximate PPM and its sensitivity are used to linearize performance

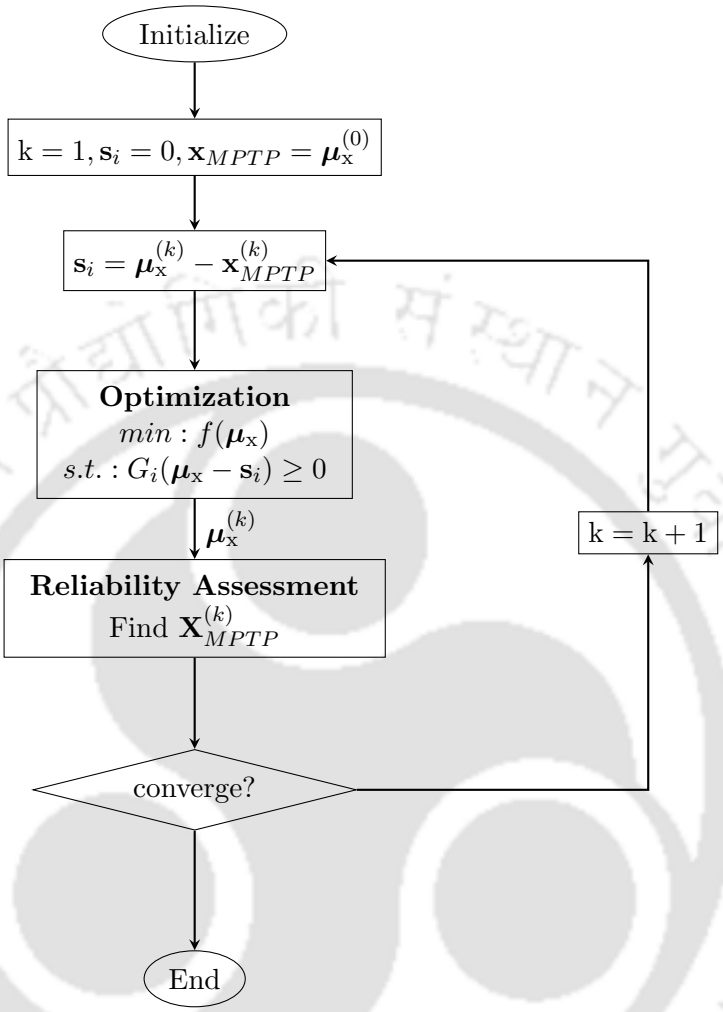


Figure 2.5: Flowchart of decoupled-loop method

function using Taylor series expansion.

An sequential approximate programming (SAP) method was developed by Cheng et al. (2006) that uses a sub-programming problem and approximates the objective function and performance functions. Taylor series expansion is used to approximate the reliability constraints at each sub-programming step. Furthermore, the approximated reliability index (β) is obtained by recursive formula based on optimality condition at MPP that is represented by equation (2.25).

$$\beta_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}) = \hat{\beta}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)}) + (\nabla_{\boldsymbol{\mu}_{\mathbf{X}}} \hat{\beta}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)}))^T (\boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \boldsymbol{\mu}_{\mathbf{X}}^{(k-1)}), \tag{2.25}$$

where $\hat{\beta}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)})$ is the approximate reliability index and its derivative is represented by $\nabla_{\boldsymbol{\mu}_{\mathbf{X}}}\hat{\beta}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)}) = \frac{\nabla_{\boldsymbol{\mu}_{\mathbf{X}}}G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)})}{\|\nabla_u G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)})\|}$, $\boldsymbol{\mu}_{\mathbf{X}}^{(k)}$ is the mean values of random variable at $k-$th iteration. The approximate reliability index is calculated as

$$\begin{aligned}
\hat{\beta}(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}) = &\quad \lambda_i^{(k-1)}[G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)}) - (\mathbf{U}^{(k-1)})^T\nabla_u G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)})],\\
\lambda_i^{(k-1)} = &\quad \frac{1}{\|\nabla_u G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)})\|},\\
\mathbf{U}^{(k)} = &\quad -\hat{\beta}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)})\lambda_i^{(k-1)}\nabla_u G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)},\mathbf{U}^{(k-1)}),
\end{aligned} \tag{2.26}$$

where $\lambda_i^{(k-1)}$ is the control factor to estimate reliability index iteratively, $\nabla_u G_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k-1)}$ is the gradient estimated with respect to random variable in the standard normal space $\mathbf{U}$.

A hybrid sequential approximate programming (HSAP) method was proposed by Meng et al. (2017) in which SAP and SORM are combined. It is developed by a distance checking criterion and convex approximate method. This criterion can identify the feasibility of the probabilistic performance function. The convex approximate method is constructed by sensitivity and function value of probabilistic performance function, and thus, the computational efficiency is improved. The distance checking criterion is given as

$$\begin{aligned}
|\hat{\beta}_i^{(k)} - \beta_i^t|/|\beta_i^t| \leq \varepsilon, &\quad \text{if active,}\\
|\hat{\beta}_i^{(k)} - \beta_i^t|/|\beta_i^t| > \varepsilon, &\quad \text{otherwise,}
\end{aligned} \tag{2.27}$$

where $\hat{\beta}_i^{(k)}$ is the approximate reliability index that is calculated at $k-$th iteration and equation (2.27) represents active and inactive nature of performance function. SORM based reliability index is calculated if the criterion is true. Otherwise, SAP is performed to calculate $\mathbf{U}^*$.

A decoupled-loop method was proposed by Li et al. (2010) in which the loops are disintegrated and performed sequentially. Deterministic optimization is performed to locate the mean of uncertain design variable, and reliability analysis is used to estimate MPP. Thereafter, a penalty is added to each of the performance functions for improving the accuracy without linearising or transforming the performance function. The violated deterministic performance function is shifted as given in equation (2.28) to reduce the error and thereby, the solution is driven towards feasible region.

$$\mathrm{G}_i(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}) \leq \theta_i^{(k-1)} - \alpha_i^{(k)}(\hat{\beta}_i^{(k)} - \beta_i^{(k-1)}), \tag{2.28}$$

$$\alpha_i^{(k)} = \| \frac{\theta_i^{(k-1)} - \theta_i^{(k-2)}}{\beta_i^{(k-1)} - \beta_i^{(k-2)}} \|,$$

where k represents $k-$th iteration, i refers to $i-$th performance function, and $\alpha_i^{(k)}(\hat{\beta_i^{(k)}} - \beta_i^{(k-1)})$ is the penalty term.

An adaptive decoupling approach (ADA) was proposed by Chen et al. (2013) to improve the efficiency of probabilistic optimization. It incorporates an update angle strategy which is calculated by the angle between the current MPP ($\mathbf{U}^{(k)}$) and MPP ($\mathbf{U}^{(k+1)}$) of the next iteration. This strategy is used to reduce the number of function calls. It divides the constraints into feasible, active and violated categories and then, active and violated constraints are evaluated.

Another decoupled-loop method was proposed by Torii et al. (2016) in which any reliability analysis can be employed to improve the accuracy for approximating the probabilistic performance function. The solution is estimated with least square approximation along with the shift vectors. As the method is based on RIA, large computational effort is required to solve problems with several random variables.

An optimal shifting vector (OSV) method was proposed by Chen et al. (2013a) that performs reliability analysis in the super sphere. The shifting vector is applied while estimating the performance function ($\mathrm{G}_i(\mathbf{X}) = 0$), instead of specific performance function ($\mathrm{G}_i(\mathbf{X})$). The minimum shifting coefficient that is used for solving OSV is calculated by a new reliability model, which is given in equation (2.29)

$$\begin{aligned} &\min: && \tau_i, \\ &\text{subject to} && G_i(\mathbf{U}(\boldsymbol{\gamma}) + \tau_i \mathbf{U}(\boldsymbol{\gamma}) = 0, \\ &&& i = 1, 2, \ldots, I, \end{aligned} \tag{2.29}$$

where γ is super sphere coordinate and τ_i is the shifting coefficient of $i-$th performance function.

Another method developed using the concept of shifting vector is incremental shifting vector (ISV) (Huang et al., 2016) for higher performance and better computational efficiency. It improves the convergence of RBDO method as well. The ISV ($\Delta \mathbf{S}_{\mathbf{U}}^{(k)}$) at each iteration is given in equation (2.30). This incremental vector was estimated through computation method rather than optimization, which improves the computational efficiency.

$$\Delta \mathbf{S}_{\mathbf{U}}^{(k)} = (\beta^t - \hat{\beta}^{(k)}) \left(-\frac{\nabla G(\mathbf{U})}{\|\nabla G(\mathbf{U})\|} \right). \tag{2.30}$$

All these methods performed the reliability analysis separately and improved the efficiency of RBDO methods. However, the reliability analysis itself requires many function evaluations because another optimization is required for obtaining the MPFP. To further address an issue of higher computational requirement, probabilistic performance function is approximated to deterministic performance function. These types of RBDO methods are known as single-loop method and are discussed in the following section.

2.2.5 Single-loop RBDO methods

Single-loop methods have been proposed to overcome the difficulty of computational effort required for reliability analysis. It converts the probabilistic performance functions into equivalent deterministic performance functions and thus, reliability analysis loop can be removed. The accuracy of the method is compromised by the implementation of approximate and equivalent deterministic performance function. Most of these methods use Karush Kuhn Tucker (KKT) optimality conditions in different forms to obtain an reliable solution. Figure 2.6 shows flowchart of single-loop method. It can be seen that, the complete reliability analysis loop is eliminated.

A Single-loop single vector (SLSV) (Chen et al., 1997) was proposed by transforming the probabilistic performance function into approximate deterministic performance function. The method is derived in a reduced, uncorrelated, and normalized space. An approximate MPP ($\mathbf{X}^* = \mu_{\mathbf{x}} - \beta^t \alpha^*$) is calculated on the basis of performance function sensitivities (α^*) and target reliability (β^t).

A single-loop method proposed by Kuschel and Rackwitz (1997) implements the KKT optimality conditions to identify the design point and modifies the RBDO formulation. In such manner, the inner reliability loop is avoided and a simultaneous convergence to the design variables and the design point location are achieved. Although, this method provides a single optimization loop, it requires the computation of second-order derivatives.

Another single-loop method (Liang et al., 2008) was developed for both normal and non-normal random variables that can follow distribution other than normal distribution. It imposes the KKT optimality conditions on PMA for converting inner reliability analysis loop to an equivalent deterministic performance function. The modified RBDO formulation is given in

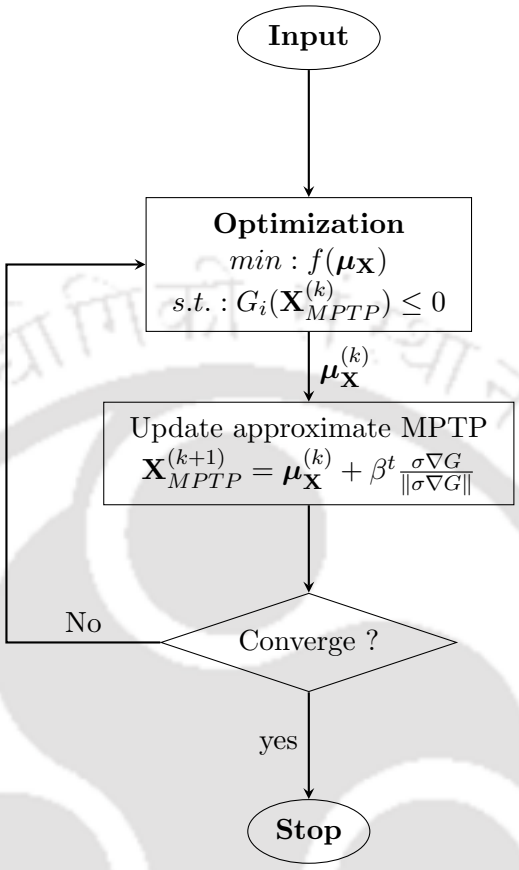


Figure 2.6: Flowchart of single-loop method

equation (2.31).

$$\begin{aligned} \min: \quad & f(\boldsymbol{\mu}_{\mathbf{x}}), \\ \text{s.t.} \quad & G_i(\mathbf{X}) \leq 0 \qquad\qquad i = 1, 2, .., I, \\ \text{where} \quad & \mathbf{X}_i = \boldsymbol{\mu}_{\mathbf{x}} - \boldsymbol{\sigma}\beta_i^t \boldsymbol{\alpha}_i, \\ & \boldsymbol{\alpha_i} = \frac{\sigma \nabla G_{i_{(\mathbf{X})}}(\mathbf{X}^{(k)})}{\|\sigma \nabla G_{i_{(\mathbf{X})}}(\mathbf{X}^{(k)})\|}, \end{aligned} \tag{2.31}$$

where $\boldsymbol{\alpha_i}$ is the search direction estimated by steepest descent search, and σ is the standard deviation. In every iteration, MPTP is estimated and simultaneously the design variable $\boldsymbol{\mu}_{\mathbf{x}}$ is changed.

Although SLSV method shows a good convergence on variety of problems, it is not found effective for dealing with highly nonlinear performance function (Aoues and Chateauneuf, 2010,

Biswas and Sharma, 2021b). Therefore, a method is improved by using conjugate gradient (CG) direction (Jeong and Park, 2017) instead of steepest descent direction. CG uses the gradient information available from the previous cycles to estimate the new MPTP. It provides successive direction vector, which tends to cross diagonally between the orthogonal steepest descent direction to improve the convergence.

A modified SLSV method was developed by Kogiso et al. (2012) using inactive and active MPTP design concept. It is combined together with modified HMV method (Youn et al., 2003). The deterministic optimum is considered as an inactive design which is taken as a staring point of RBDO. Active MPTP design corresponds to the sensitivity of the performance function evaluated at the approximate MPTP that is used as an initial searching direction of RBDO.

A semi single-loop method was proposed by Lim and Lee (2016) for efficient reliability analysis. The sensitivity analysis is used to approximate the MPTP. The approximation is validated using equation (2.32).

$$\begin{aligned} \left| \frac{\hat{\mathbf{u}}}{\|\hat{\mathbf{u}}\|}(\mathbf{n}-1) \right| &\leq \varepsilon, \\ \hat{\mathbf{u}} &= \beta^t \frac{\hat{\mathbf{u}}^{'}}{\|\hat{\mathbf{u}}^{'}\|}, \\ \hat{\mathbf{u}}^{'} &= \mathbf{u}^* + \frac{\partial \mathbf{u}^*}{\partial \boldsymbol{\mu}_{\mathbf{x}}} \left(\boldsymbol{\mu}_{\mathbf{x}}^{(k+1)} - \boldsymbol{\mu}_{\mathbf{x}}^{(k)} \right), \end{aligned} \tag{2.32}$$

where $\hat{\mathbf{u}}$ is the shifted approximate MPTP, $\hat{\mathbf{u}}^{'}$ is the approximated MPTP, $\mathbf{n}$ is the normalized steepest ascent direction of $G(\mathbf{U})$ at the approximated MPTP and parameter ε is recommended to be 10^{-4}. If the approximated MPTP is valid, the method has a complete single-loop structure. Otherwise, a fast reliability analysis is performed by selecting the shifted approximate MPTP that can be estimated by PMA.

A unilevel model was developed by Agarwal et al. (2007) in which the first-order KKT optimality conditions corresponding to probabilistic performance function are imposed in order to replace lower level optimization. This method replaces the inverse FORM problem with KKT optimality conditions. The basic difference between this method and other single-loop method is that the constraints are being presented as equality constraints.

A single-loop deterministic method (SDLM) was proposed by Shan and Wang (2008) that approximates gradient information at the MPP. It is based on the concept of reliable design space (RDS), inside which any design solution satisfies the reliability requirement. The optimization is performed within constrained RDS region. The design solution that satisfies the

reliability requirement is approximated by partial derivative of performance function at MPTP.

Another single-loop method based on statistical moments was proposed that is referred to as approximate moments approach (AMA) (Putko et al., 2001). The first-order Taylor series expansion is used to calculate the mean ($\boldsymbol{\mu}_G$) and standard deviation ($\boldsymbol{\sigma}_G$) for all performance functions $G(\mathbf{X})$ by using the expressions given in equation (2.33).

$$\begin{aligned} \boldsymbol{\mu}_G &= G(\mathbf{X}), \\ \boldsymbol{\sigma}_G^2 &\approx (\nabla G)^2 \sigma^2, \end{aligned} \tag{2.33}$$

where $\boldsymbol{\sigma}$ is the standard deviation of the input variables $\mathbf{X}$ and ∇ is the gradient operator. To replace and approximate $G(\mathbf{X}^*)$, first-order Taylor series expansion is used. In this manner, the performance function is shifted to maintain the desired reliability level. Thus, the deterministic RBDO problem can be reformulated as

$$\begin{aligned} \text{min}: \quad & f(\boldsymbol{\mu}_{\mathbf{x}}), \\ \text{s.t.} \quad & G_i(\mathbf{X}) + \beta^t \boldsymbol{\sigma}_G \leq 0, \qquad i = 1, 2, .., I, \\ & \boldsymbol{\mu}_{\mathbf{X}} - \beta^t \boldsymbol{\sigma} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}} + \beta^t \boldsymbol{\sigma}. \end{aligned} \tag{2.34}$$

It can be noted that all single-loop methods aim to accelerate the convergence by introducing approximations for performing reliability analysis. Most of the methods propose equivalent deterministic performance function while others estimate approximate MPP. Therefore, the results obtained by these methods may not be as accurate as double-loop and decoupled-loop methods.

All the methods discussed in this section are based on gradient-based optimization which are susceptible to converge to a locally optimal and reliable solution. Therefore, various meta-heuristic algorithms are used for solving RBDO problems to obtain global solution. In the next section, literature related to different types of meta-heuristic algorithms are discussed.

2.2.6 Meta-heuristic algorithms for RBDO

The literature survey presented earlier used gradient-based numerical optimization methods for obtaining the optimal solution. As these methods depend on gradient information, they often get trapped in the region of local optima. To overcome this issue, global optimization techniques like Genetic algorithms (GA) (Goldberg, 1989), particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), differential evolution (DE) (Storn and Price, 1997), and artificial bee colony optimization (ABC) (Karaboga and Basturk, 2007) are coupled with RBDO to obtain

global reliable solution.

A GA-based RBDO method was proposed by Shayanfar et al. (2014) for solving structural problems through a software application. It employs GA as an optimization technique in which probabilistic performance functions are evaluated using finite element reliability analysis modules offered by OpenSees software. It is developed to deal with both discrete and continuous design variables.

An inverse reliability analysis using PMA coupled with DE (Lobato et al., 2017) was used to find the global reliable solution. It has implemented neighbourhood exploring evolution strategy to explore the search space. DE is performed on the outer loop and PMA is used in the inner loop of double-loop method. The penalty function method is used for establishing fitness value. DE is improved by a mutation scheme using the $p-$best strategy, and a new modification scheme for scaling factor and crossover rate parameter is also proposed. The reliability analysis is performed using improved Latin hypercube sampling and effective importance sampling (Truong and Kim, 2018).

A machine learning-based reliability optimization approach by fusing ABC (Zhu et al., 2020) was proposed in which traditional support vector regression (SR) model, and multi-population genetic algorithm (MPGA) are used. ABC algorithm is applied to find the optimal parameters in the SR model to establish accurate improved support vector regression method (ISRM). Then, MPGA is implemented with the reliability optimization model to obtain the global solution.

In another study, a FORM-based RBDO problem was solved using ABC (Zhao et al., 2016) for obtaining reliable solution for geotechnical engineering system. The fitness is estimated with the penalty function method, and the RBDO structure is based on double-loop method.

A joint optimization framework was proposed by Chen et al. (2022) in which two-step ABC optimization, and ABC combinatorial optimization are coupled. It is developed with a Max-Min reliability optimization that includes both sub-carrier assignment and power allocation for reliability analysis. The ABC algorithm is creatively invoked as a resource allocation technique to guarantee the optimal solution.

An enhanced weighted simulation-based design method (Safaeian Hamzehkolaei et al., 2016) was proposed with PSO using double-loop structure. The samples of weighed simulation method (WSM) are considered as initial population of the PSO. Then, the same population is employed for estimating the safety level of each PSO swarm. This strategy helps in preventing reliability analysis of design candidates. In addition, a shift strategy is also introduced to

increase the capability of the WSM.

Yang and Hsieh (2013) proposed a RBDO method that combines PSO, Support Vector Machine (SVM), and Subset Simulation (SS) together to deal with discrete design parameters and non-differentiable performance functions. SS is used for estimating small failure probabilities. SVM is adopted for performing reliability analysis of each candidate solutions, and PSO is used to solve the discrete optimization problem. This hybrid strategy is mutually beneficial since the SVM classifier can help PSO to evaluate the feasibility of solutions with high efficiency. The optimal solution assists in attaining better accuracy for the SVM classifier.

Apart from the double-loop methods, the decoupled-loop methods have also used with meta-heuristic algorithms. An improved DE coupled with SORA (Khodam et al., 2021) was used to deal with both continuous and discrete variables for solving RBDO problems. DE is made adaptive using a selection scheme for mutation, and elitist selection for next generation target vectors. The reliability analysis is performed using modified chaos control method.

Another method based on PSO (Elegbede, 2005) was proposed for structural reliability assessment. The penalty method is used with relaxed RIA to estimate the reliability index. The accuracy of obtaining reliability index is compared with response surface method and complete quadratic response surface with re-sampling method, which is found to be better.

A nested loop was employed to solve RBDO problems that used multi-objective PSO (Zafar et al., 2020) in the outer-loop. The inner-loop is performed by surrogate modelling with MCS sampling. The mean and coefficient of variation are taken as two objective functions that need to be optimized for considering reliability and robustness simultaneously. An adaptive multiple Kriging models are built for problems with multiple failure modes to predict the reliability.

A method was suggested by Gargama et al. (2014) in which the design points were created throughout the design space, and the probability satisfactory factor (PSF) is calculated at each point using Monte Carlo simulation. These design vectors and their corresponding PSFs were then utilized to train an artificial neural network (ANN). Once trained, the ANN is employed in the design optimization process to determine the desired PSF while adhering to the constraints of the design variables. Next, a real-coded genetic algorithm (GA) is utilized with a simulated binary crossover operator, a parameter-based mutation approach, and a parameter-free penalty function approach for obtaining reliable solutions.

A research paper addressed the deterministic and probabilistic optimization of the nailing system by employing recently introduced atomic orbital search algorithms (Shirgir et al., 2023)

used for optimization purposes. The first-order reliability method is applied to conduct reliability analysis which is again based on double-loop method. The design variables in the optimization process include the nailing system's geometric properties, while the soil's mechanical properties are considered as random variables.

A unified framework (Meng et al., 2022) was used to enhance the performance of RBDO algorithms for complex problems. The framework introduces three strategies: a generalized decoupling evolutionary and metaheuristic RBDO framework, a particle's memory saving strategy, and an adaptive fractional-order equilibrium optimizer algorithm. for solving the problem This algorithm enhances the search efficiency and global convergence capacity of the RBDO algorithm.

A method based on whale optimization algorithm (Jafari-Asl et al., 2021) was also developed for obtaining reliable design of labyrinth spillway. In this case, MCS and hybrid Artificial Neural Network (ANN) are coupled and used for reliability analysis.

Both double-loop and decouple-loop methods incorporate complete reliability analysis and therefore, these methods are computationally expensive. To overcome this issue, single-loop methods using meta-heuristic algorithms (Ho-Huu et al., 2018) have also been developed. An improved DE is proposed by using modified roulette wheel selection with stochastic acceptance in order to sort out solutions for mutation. After performing mutation, an elitist selection technique is used for picking solutions for next generation target vectors of DE (Ho-Huu et al., 2018).

In another attempt, a single-loop RBDO method was developed using PSO (Liao and Ivan, 2014). To improve the efficiency for extracting the gradient information, the multi-variable constraint is transformed to a single-variable constraint using exponential polynomial coefficient to represent a probability density function. The estimated reliability and its derivative are then incorporated with PSO for obtaining optimal solution.

From the literature, it can be found that most of the real world problems consist of multiple objectives apart from uncertainty. In the following section, the literature related to the meta-heuristic algorithms for multi-objective RBDO are discussed which were used to generate reliable Pareto-optimal solutions.

2.3 Multi-objective reliability-based design optimization

Multi-objective optimization problems consist of multiple conflicting objective functions, i.e., improvement of one objective comes at the expense of another objective. Unlike single-objective problems, multi-objective problem generates a set of optimal solutions that are known as Pareto-optimal solutions. These Pareto-optimal solutions are a set of trade-off solutions (also called non-dominated solutions) that are optimal in the sense that no other solutions in the design space can dominate them when all the objectives are considered. The multi-objective RBDO (MORBDO) problems are solved to generate similar Pareto-optimal solutions that are also reliable. These reliable solutions provide the designer a choice to select a solution as per the requirement and weights given to the objective functions. A simple MORBDO formulation (Deb et al., 2009, Li et al., 2015, Hamzehkolaei et al., 2018) can be written as

$$\begin{array}{lll} \text{Minimize} & f_m(\boldsymbol{\mu}_{\mathbf{X}}), & m = 1, \ldots, M, \\ \text{subject to} & P[G_i(\mathbf{X}) \leq 0] \leq P_{f_i}^T = \Phi(-\beta_i^T), & i = 1, \ldots, I, \\ & \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{L})} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{U})},\ \mathbf{X}^{(\mathrm{L})} \leq \mathbf{X} \leq \mathbf{X}^{(\mathrm{U})}, & \end{array} \tag{2.35}$$

where $f_m(\cdot)$ is the $m-$th objective function that is written using mean value ($\boldsymbol{\mu}_{\mathbf{X}}$) of random variable ($\mathbf{X}$).

A reliability-based multi-objective optimization problem was solved by Deb et al. (2009) using non-dominated sorting genetic algorithm (NSGA-II). The computation for RIA is made faster for reliability analysis by using an intermediate MPP with unit reliability index ($\beta^t = 1$). Unidirectional search is performed based on Newton-Raphson approach to locate MPP on $G_i(\mathbf{U}) = 0$. Due to the conversion of multivariable to single-variable problem, the method is quite fast. Multiple constraints are handled by the system reliability concept and using Ditlevsen's bounds to compute a more accurate probability of failure.

A multi-objective differential evolution (MODE) was developed by Lobato et al. (2017) based on double-loop structure. The reliability analysis is performed in the inner loop by PMA and the Pareto-optimal solutions are estimated by MODE in the outer-loop. The mechanisms of rank ordering and neighborhood potential solution candidates are utilised for exploring the search space.

In another study, to solve reliability-based robust design optimization (RBRDO) a method was developed by coupling different methods and strategies. The original objective function is represented with two new objectives that are the maximization of the robustness parameter and the maximization of the reliability coefficient. This problem is solved by MODE (Lobato

et al., 2017) to generate potential candidates and evaluating each candidate by effective mean concept approach. Thereafter, each candidate is evaluated considering the inverse reliability analysis strategy and finally, the non-dominated solutions are generated.

A multi-objective and multi-case RBDO was developed by Sun et al. (2017) in which radial basis function is used for approximating the responses of performance function and objective functions. NSGA-II and MCS are employed for generating optimal reliable solutions. Sun et al. (2017) also proposed a surrogate-modelling technique that is used to estimate the responses of performance function. A multi-objective PSO is incorporated to obtain optimal reliable Pareto-optimal solutions that are coupled with MCS for reliability analysis.

A multi-objective RBDO (Sleesongsom and Bureerat, 2020) was solved by combining real-code population using incremental learning, DE with worst-case scenario, and fuzzy sets for reliability analysis. As this technique involves worst-case and fuzzy sets for estimating reliability, it comes under the group of non-probabilistic approaches where the intervals of uncertainties without accurate probability distributions are specified.

A multi-objective RBDO was proposed by Li et al. (2015) based on double-loop method. In the outer loop, NSGA-II is implemented to generate the robust Pareto-optimal solutions. In the inner loop, MCS is performed for evaluating the impact responses of the mixed uncertainties that were considered during design.

A multi-objective RBDO method was proposed by Dammak and Hami (2020) using constrained NSGA-II that is coupled with kriging meta-model for reliability analysis using surrogate method. A minimum distance selection method is used to find the best point from the Pareto-optimal front with regard to the reliability constraint.

A multi-objective reliability-redundancy allocation problem (MORRAP) was proposed by Muhuri et al. (2018) that is formed using a novel formulation that effectively handles higher-order uncertainties using interval type-2 fuzzy sets. The uncertainties were handled by interval type-2 fuzzy multi-objective reliability redundancy allocation problem. It considers multiple empirical and optimizes reliability, cost, and weight goals for a series-parallel system using interval type-2 fuzzy numbers..

A hybrid reliability-based design optimization method was proposed by Liu et al. (2023) considering the presence of both random and interval variables. It was developed using a decoupled method that updates the adaptive Kriging model for performance measure function approximation. It also integrates hybrid inverse reliability analysis, performance measure function approximation, and equivalent deterministic optimization. The two-stage enrichment of

the Kriging model allows precise approximation of performance measure functions within the region of interest.

An evolutionary multi-objective genetic algorithm was proposed by Forouzandeh Shahraki and Noorossana (2014) to find the reliable and robust Pareto-optimal solutions for solving RBRDO problems. The reliability analysis is performed by PMA. The process capability index is used to obtain the best solution for implementation, which is verified with MCS.

A cell evolution method was developed by Chen et al. (2014) in which Sobol's quasi-random sequence and spherical parameterization method is used to create template cells. This template cells helps in reliability analysis by generating MPP easily without gradient calculation. In recent studies, time-dependent RBRDO (Yu et al., 2019) has been solved using evolutionary algorithm. NSGA-III is used for optimization and dimension reduction method is used to construct an extreme value model using the sparse grid-based stochastic collocation method for time-dependent reliability analysis. Celorrio and Patelli (2021) proposed Bayesian multi-objective RBDO to consider different types of uncertainties. The optimization is performed by multi-objective PSO and Bayesian interference is used for reliability analysis.

2.4 Closure

It can be seen from the above literature survey that RBDO methods are used to obtain reliable solutions for problems involving uncertainties. The uncertainties are considered in the optimization problem with the help of probabilistic performance function. Reliability analysis is performed to solve the probabilistic performance function. There exists various methods for solving reliability analysis, such as simulation based methods, analytical methods, approximate integration methods, to name a few. Among them, analytical methods are widely used due to their computational efficiency. They are broadly classified as double-loop methods, decoupled-loop methods, and single-loop methods. These methods consist of two parts, i.e., a) optimization and b) reliability analysis. The performance of these methods depends on the accurate estimation of MPP which is obtained by reliability analysis. In double-loop method, a nested-loop optimization is used to estimate the mean values of random variables in the outer-loop, whereas the inner loop is used for the MPP estimation of the performance functions. In decoupled-loop method, optimization and reliability analysis are performed sequentially. Both these methods involve complete reliability analysis, making them computationally expensive. On the other hand, singe-loop methods approximate reliability analysis and thus the computa-

tional efficiency improves. These methods are found effective in generating a reliable solution efficiently.

It has been observed that single-loop methods often face the problem of convergence while obtaining MPP and starts oscillating for concave and highly non-linear performance functions. There exists a trade-off between the accuracy and efficiency, as observed from the literature. It has been seen that there is no proper criterion to track the oscillation among MPPs. Although chaos control theory shows significant improvement for controlling oscillation in double-loop methods, proper implementation with single-loop methods has to be explored. For obtaining global solution, meta-heuristic algorithms are combined with double-loop and decoupled-loop methods, which demonstrate effective generation of reliable solution. As these methods require high computational cost, single-loop methods can be the right choice to use with meta-heuristic algorithms for obtaining global solution. It has also been observed that most of the practical problems consist of multiple objectives, and obtaining reliable Pareto-optimal solutions efficiently is a big challenge. Therefore, development of single-loop multi-objective optimization algorithms for solving RBDO problems can also be studied. These bottlenecks in the literature lead to a scope of developing single-loop methods for single and multi-objective optimization problems in this thesis.

Chapter 3

Development of single-loop method

The first objective of the thesis is to develop an accurate and efficient single-loop method for solving RBDO problems. A FORM-based single-loop method is coupled with the shifting vector approach of SORA (Du and Chen, 2004) to achieve this objective. The accuracy of the method is improved by implementing the KKT optimality conditions to approximate MPTPs of each performance function. Moreover, conjugate gradient search direction is used to update MPTP in normal space. To improve computational efficiency, shifting vector is incorporated with all performance functions, which gets updated at every iteration.

In the following sections, a single-loop approach and SORA are discussed. Thereafter, the proposed method is discussed in detail along with flowchart and implementation steps.

3.1 Single-loop approach

The single-loop approach (SLA) approximates the reliability analysis of double-loop method and establishes an equivalent single-loop deterministic optimization method. In this thesis, Karush-Kuhn Tucker (KKT) optimality conditions are used in equation (2.8), which are given in a vector form as

$$\nabla G(\mathbf{U}) - \lambda \nabla H(\mathbf{U}) = 0, \tag{3.1}$$

where λ is the Lagrange multiplier, and $H(\mathbf{U}) = \|\mathbf{U}\|^2 - \beta_i^{t2}$ after squaring both sides of equality constraint of equation (2.8). Using equation (3.1) and $\nabla H(\mathbf{U}) = 2\mathbf{U}$ yields

$$\nabla G(\mathbf{U}) - 2\mathbf{U}\lambda = 0. \tag{3.2}$$

After simplification, $\mathbf{U}$ can be written as $\frac{\nabla G(\mathbf{U})}{2\lambda}$. After multiplying $\|\nabla G(\mathbf{U})\|$ in the numerator and denominator of the right hand side of $\mathbf{U}$, we get

$$\mathbf{U} = \frac{\nabla G(\mathbf{U})}{2\lambda}\frac{\|\nabla G(\mathbf{U})\|}{\|\nabla G(\mathbf{U})\|} = \frac{\|\nabla G(\mathbf{U})\|}{2\lambda}\frac{\nabla G(\mathbf{U})}{\|\nabla G(\mathbf{U})\|} = \beta\boldsymbol{\alpha}, \tag{3.3}$$

where $\boldsymbol{\alpha} = \frac{\nabla G(\mathbf{U})}{\|\nabla G(\mathbf{U})\|}$ is the unit direction, and $\beta = \frac{\|\nabla G(\mathbf{U})\|}{2\lambda}$ is a constant at the optimal solution $\mathbf{U}^*$. The gradient calculation is performed in the $U-$space, but the random variables lie in the $X-$space. Therefore, the $X-$space is transformed to the $U-$space using the following relationship.

$$\mathbf{X} = \boldsymbol{\mu}_{\mathbf{X}} + \boldsymbol{\sigma}_{\mathbf{X}}\mathbf{U}, \tag{3.4}$$

where $\sigma_{\mathbf{X}}$ is the standard deviation of $\mathbf{X}$. Substituting $\mathbf{U}$ from equation (3.3) in equation (3.4) and using chain rule, we get approximate MPTP in the $X-$space as

$$\mathbf{X}_{MPTP} = \boldsymbol{\mu}_{\mathbf{X}} + \boldsymbol{\sigma}_{\mathbf{X}}\beta\boldsymbol{\alpha} = \boldsymbol{\mu}_{\mathbf{X}} + \boldsymbol{\sigma}_{\mathbf{X}}\beta\frac{\boldsymbol{\sigma}_{\mathbf{X}}\nabla G_{\mathbf{X}}(\mathbf{d},\mathbf{X})}{\|\boldsymbol{\sigma}_{\mathbf{X}}\nabla G_{\mathbf{X}}(\mathbf{d},\mathbf{X})\|}, \tag{3.5}$$

where $\mathbf{X}_{MPTP}$ is the MPTP of the random variable $\mathbf{X}$. Using the expression of MPTP, the single-loop formulation can be written as

$$\begin{aligned}
&\text{Min. } f(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{X}}),\\
&\text{s.t.: } G_i(\mathbf{d}, \mathbf{X}^{(k)}_{i,MPTP}) \leq 0, \qquad\qquad i = 1, \ldots, I,\\
&\text{where } \mathbf{X}^{(k)}_{i,MPTP} = \boldsymbol{\mu}^{(k)}_{\mathbf{X}} + \beta^t_i\boldsymbol{\sigma}_{\mathbf{X}}\boldsymbol{\alpha}^{(k)}_{i,\mathbf{X}},\\
&\qquad \boldsymbol{\alpha}^{(k)}_{i,\mathbf{X}} = \frac{\boldsymbol{\sigma}_{\mathbf{X}}\ \nabla G_{i,\mathbf{X}}(\mathbf{d}^{(k)}, \mathbf{X}^{(k-1)}_{i,MPTP})}{\|\boldsymbol{\sigma}_{\mathbf{X}}\ \nabla G_{i,\mathbf{X}}(\mathbf{d}^{(k)}, \mathbf{X}^{(k-1)}_{i,MPTP})\|},\\
&\qquad \boldsymbol{\mu}^{(\mathrm{L})}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}^{(\mathrm{U})}_{\mathbf{X}}, \mathbf{d}^{(\mathrm{L})} \leq \mathbf{d} \leq \mathbf{d}^{(\mathrm{U})},
\end{aligned} \tag{3.6}$$

where $\boldsymbol{\alpha}^{(k)}_{i,\mathbf{X}}$ is the normalised gradient vector or steepest descent direction of the performance function with respect to random variable ($\mathbf{X}$). It is to be noted that $\boldsymbol{\alpha}^{(k)}_{i,\mathbf{X}}$ is calculated using $\boldsymbol{\sigma}_{\mathbf{X}}$ and ∇G at MPTP of the last iteration.

3.1.1 Sequential optimization and reliability assessment

SORA (Du and Chen, 2004) was developed on the platform of decoupled-loop RBDO method, in which the optimization and the reliability assessment were performed sequentially. The MPTP was estimated by PMA for each probabilistic functions. The "shifting vector" $\mathbf{S}^{(k+1)}_i$

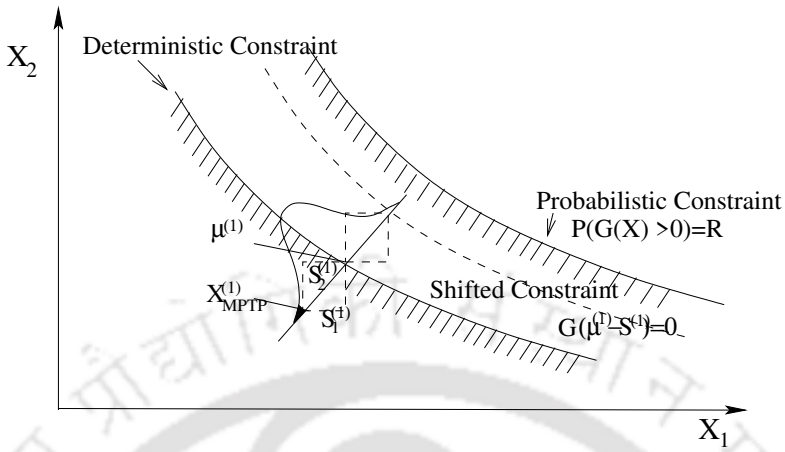


Figure 3.1: Shifting vector approach

was used to push the violated deterministic constraint toward the feasible direction. The shift vector was calculated as

$$\mathbf{S}_i^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \mathbf{X}_{\text{MPTP}}^{(k)}, \tag{3.7}$$

where $\mathbf{X}_{\text{MPTP}}^{(k)}$ is the MPTP estimated by PMA. Deterministic optimization was performed with the shifted constraints until convergence crterian was satisfied. This method used the reliability information from the previous cycle to shift the violated constraint.

Figure 3.1 shows the schematic diagram of shifting vector approach. The generation of $\mathbf{S}_i^{(k)}$ is an iterative process which helps in estimating the feasibility of the constraints until the reliability is satisfied.

3.2 The proposed single-loop shifting vector method with conjugate gradient search (SLShV-CG)

In SLA, the reliability analysis is performed by approximating the MPTP using KKT conditions. Since, the updation of MPTP requires gradient informations, SLA shows inefficiency and inaccuracy for RBDO problems with highly non-linear constraints. On the other hand, SORA decouples optimization and reliability analysis and improves computational efficiency while solving RBDO problems with any type of constraints. Therefore, SLA is coupled with the shifting vector approach of SORA, in which the constraints with less reliability is shifted toward the feasible direction. In this method, the constraints are not required to be approximated

using Taylor series. The proposed method is explained in the following subsections.

3.2.1 Details of SLShV-CG method

Since the shift vector approach of SORA is coupled with single-loop approach, the RBDO formulation based on SLA is modified as

$$\begin{aligned} &\text{min. } f(\boldsymbol{\mu}_{\mathrm{x}}), \\ &\text{s.t.: } G_i(\boldsymbol{\mu}_{\mathrm{x}}^{(k)} - \mathbf{S}_i^{(k)}) \leq 0, \qquad i = 1, 2, \ldots, I, \\ &\mathbf{S}_i^{(k)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \mathbf{X}_{MPTP}^{(k)}. \end{aligned} \tag{3.8}$$

The shifting vector $\mathbf{S}_i^{(k)}$ gets updated in every iteration using the approximate MPTP, which is given as

$$\mathbf{X}_{MPTP}^{(k)} = \boldsymbol{\mu}_{\mathbf{x}}^{(k)} + \beta_i^t \boldsymbol{\sigma}_{\mathbf{x}} \boldsymbol{\alpha}_i^{(k-1)}, \tag{3.9}$$

Note that the vector $\boldsymbol{\sigma}_{\mathbf{x}}$ is multiplied with the constraint direction vector $\boldsymbol{\alpha}_i^{(k-1)}$. This multiplication is performed by multiplying each component of vector $\boldsymbol{\sigma}_{\mathbf{x}}$ with the corresponding component of vector $\boldsymbol{\alpha}_i^{(k-1)}$. The direction $\boldsymbol{\alpha}_i^{(k-1)}$ is found using the conjugate gradient direction, which is given as

$$\begin{aligned} \boldsymbol{\alpha}_i^{(k-1)} &= \frac{\mathbf{D}_i^{(k-1)}}{\|\mathbf{D}_i^{(k-1)}\|}, \\ \mathbf{D}_i^{(k-1)} &= \boldsymbol{\sigma}_{\mathbf{x}} \nabla G_i^{(k-1)} + \frac{(\boldsymbol{\sigma}_{\mathbf{x}} \nabla G_i^{(k-1)})^T (\boldsymbol{\sigma}_{\mathbf{x}} \nabla G_i^{(k-1)})}{(\boldsymbol{\sigma}_{\mathbf{x}} \nabla G_i^{(k-2)})^T (\boldsymbol{\sigma}_{\mathbf{x}} \nabla G_i^{(k-2)})} \cdot \mathbf{D}_i^{(k-2)}, \end{aligned} \tag{3.10}$$

where $\mathbf{D}_i^{(k)}$ signifies the direction vector of $i-$th constraint at the $k-$iteration. Since CG direction is used to update the MPTP of the proposed hybrid single-loop method with the shifting vector approach of SORA method, it is referred to as SLShV-CG. Following are the steps of SLShV-CG.

Step 1 Set $k = 0$, initial design variables $\mathbf{X}^{(0)} = \boldsymbol{\mu}_{\mathbf{x}}^{(0)}$, $\mathbf{X}_{MPP}^{(0)} = \boldsymbol{\mu}_{\mathbf{x}}^{(0)}$ and the given standard deviation $\boldsymbol{\sigma}_{\mathbf{x}}$ and target reliability index β^t.

Step 2 Calculate the shifting vector of the $i-$th constraint as

$$\mathbf{S}_i^{(k)} = \boldsymbol{\mu}_{\mathbf{x}}^{(k)} - \mathbf{X}_{MPTP}^{(k)}. \tag{3.11}$$

The initial shifting vector will be zero as $\mathbf{X}_{MPTP}^{(0)} = \boldsymbol{\mu}_{\mathbf{x}}^{(0)}$. Therefore, the first optimization

will be a deterministic evaluation of optimal solution.

Step 3 Perform deterministic optimization of the design problem with the shifted constraints.

$$\begin{aligned} &\min\ f(\boldsymbol{\mu}_{\mathrm{x}}),\\ &\text{s.t.: } G_i(\boldsymbol{\mu}_{\mathrm{x}}^{(k)} - \mathbf{S}_i^{(k)}) \le 0, \qquad i = 1, 2, \ldots, I,\\ &\quad \mathbf{S}_i^{(k)} = \boldsymbol{\mu}_{\mathbf{x}}^{(k)} - \mathbf{X}_{MPTP}^{(k)}. \end{aligned} \tag{3.12}$$

Step 4 Calculate the conjugate direction vector

$$\boldsymbol{\alpha}_i^{(k)} = \begin{cases} \dfrac{\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i}{\|\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i\|}\Big|_{\boldsymbol{\mu}_{\mathbf{x}}^{(0)}}, & \text{if } k = 0,\\ \dfrac{\mathbf{D}_i}{\|\mathbf{D}_i\|}\Big|_{\mathbf{X}_{MPTP}^{(k)}}, & \text{otherwise,} \end{cases} \tag{3.13}$$

where

$$\mathbf{D}_i^{(k)} = \begin{cases} \boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i, \quad k \le 1\\ \boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i^{(k)} + \dfrac{(\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i^{(k)})^T(\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i^{(k)})}{(\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i^{(k-1)})^T(\boldsymbol{\sigma}_{\mathbf{x}}\nabla G_i^{(k-1)})} \cdot \mathbf{D}_i^{(k-1)}, \quad \text{otherwise.} \end{cases}$$

Step 5 Set $k = k + 1$ and update $\mathbf{X}_{MPTP}^{(k)}$ in the original space as

$$\mathbf{X}_{MPTP}^{(k)} = \boldsymbol{\sigma}_{\mathbf{x}}^{(k)} + \beta^t \boldsymbol{\sigma}_{\mathbf{x}} \boldsymbol{\alpha}_i^{(k-1)}. \tag{3.14}$$

Step 6 If the convergence criterion
$\|f(\boldsymbol{\mu}_{\mathbf{x}}^{(k+1)}) - f(\boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| / \|f(\boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| \le 0.001$
or $\|\boldsymbol{\mu}_{\mathbf{x}}^{(k+1)} - \boldsymbol{\mu}_{\mathbf{x}}^{(k)}\| \le 0.001$ is satisfied, terminate. Otherwise, go to Step 2.

The flowchart of SLShV-CG is shown in Figure 3.2 in which all the steps are shown. It can be observed that SLShV-CG does not require transformation from the original variable space to the standard normal variable space.

3.3 Numerical examples

In this section, four mathematical and four engineering RBDO examples are solved to demonstrate the accuracy and computational efficiency of the proposed SLShV-CG method. The accuracy of the obtained solutions is evaluated through Monte-Carlo Simulations (MCS) using

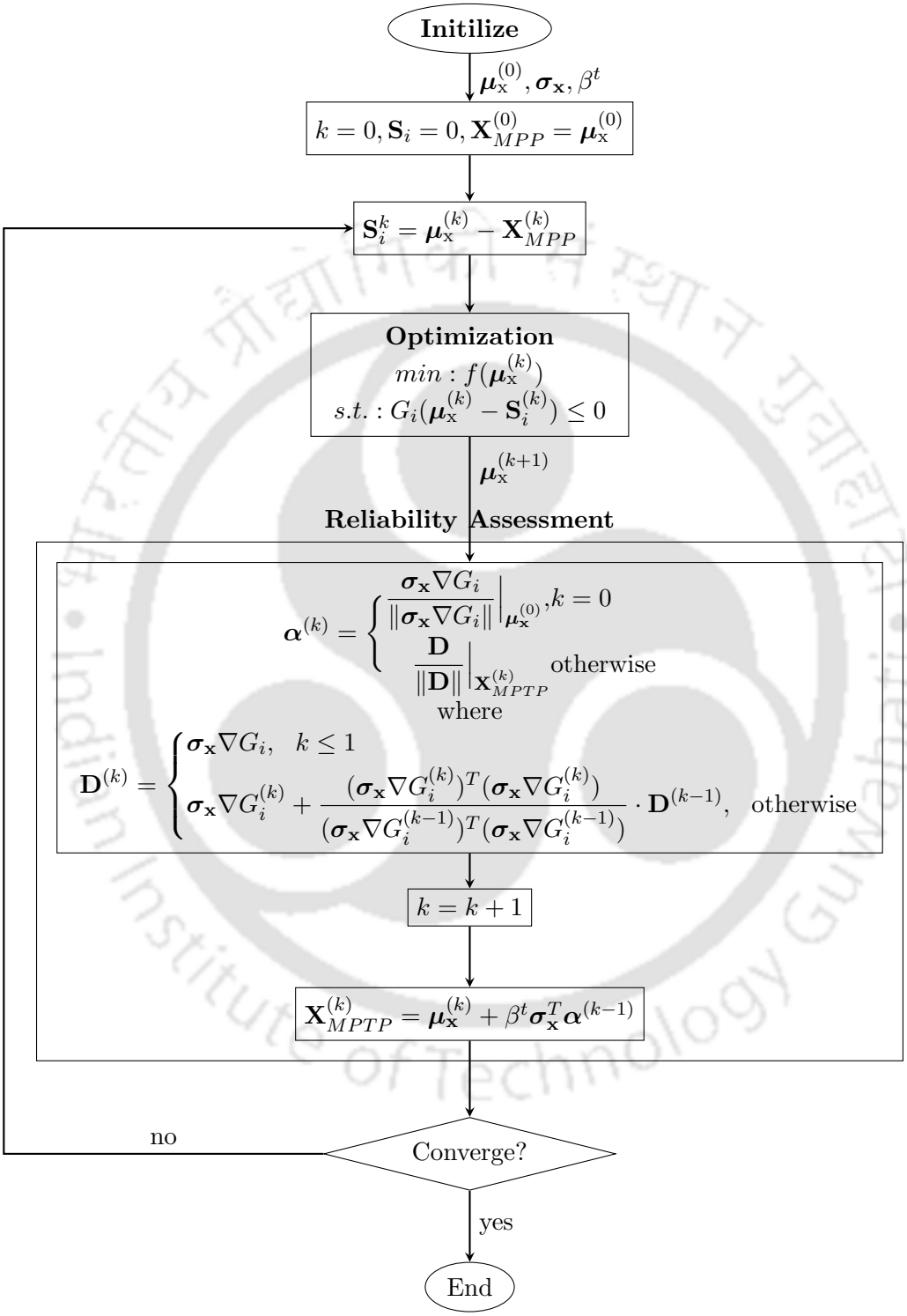


Figure 3.2: Flowchart of SLShV-CG

one million samples and the target reliability index β_{MCS}^t for each probabilistic constraint is determined. The objective function value and the number of function evaluations are used to evaluate the performance of the algorithms. The computational efficiency is measured through the "Number of Function Calls" (NFC) in which f_{FC} is the measure of objective function calls and g_{FC} is the measure of probabilistic function calls. 'Iter' is used to denote the total number of iterations required by the optimization algorithm for convergence. The performance of SLShV-CG is also compared with that of PMA-AMV (Tu et al., 1999), PMA-CGA (Ezzati et al., 2015), SLSV (Liang et al., 2008), SLSV-CG (Jeong and Park, 2017), SORA (Du and Chen, 2004) and ASORA Yi et al. (2016). It is noted that all methods are initialized with the same initial point and are terminated using the same termination conditions given at Step 6 of the SLShV-CG algorithm. For optimization, the tool "fmincon" of MATLAB is used as an optimizer.

3.3.1 Example 1

The first example (Jiang et al., 2017a) as given in equation (3.15) is a highly non-linear mathematical example. The constraint $G_2(\mathbf{X})$ is concave in nature, while $G_1(\mathbf{X})$ and $G_3(\mathbf{X})$ are convex in nature. The initial point is taken as $\boldsymbol{\mu}_{\mathbf{x}}^{(0)} = [5.0, 5.0]^T$ and the reliability for each constraint is increased to $\beta_i^t = 3.5$.

Table 3.1 presents the optimal solutions obtained from the methods. The target reliability achieved by the solutions for each probabilistic constraint is shown from the sixth column to the eighth column, which is evaluated through MCS. The constraint function calls $\mathrm{g_{FC}}$ denote the total number of times the constraint function is called by "fmincon" ($g_{fmincon}$), plus the number of times it is called for its gradient calculation (grf). Meaning, $\mathrm{g_{FC}}$ is equal to $g_{fmincon} + grf \times nc \times nv \times 2$, where nc and nv represent the number of constraints and the number of variables, respectively and the numeric 2 is multiplied as the central difference method is used for gradient calculation. Similarly, the number of the cost function calls (f_{FC}) is calculated by determining the number of function called by "fmincon".

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1},\ \mu_{x_2}]^T \\
&\text{min: } -\frac{(\mu_{x_1}+\mu_{x_2}-10)^2}{30}-\frac{(\mu_{x_1}-\mu_{x_2}+10)^2}{120}, \\
&\text{s.t.: } Pr\left[G_1(\mathbf{X})=1-\frac{x_1^2x_2}{20}>0\right]\leq\phi(-\beta_1^t), \\
&\quad Pr[G_2(\mathbf{X})=-1+(Y-6)^2+(Y-6)^3 \\
&\quad -0.6(Y-6)^4+Z>0]\leq\phi(-\beta_2^t), \\
&\quad Pr\left[G_3(\mathbf{X})=1-\frac{80}{(x_1^2+8x_2+5)}>0\right]\leq\phi(-\beta_3^t), \\
&\quad Y=0.9063x_1+0.4226x_2, \\
&\quad Z=0.4226x_1-0.9063x_2, \\
&\quad 0\leq\boldsymbol{\mu}_{\mathbf{x}_l}\leq 10,\ x_l\sim N(\boldsymbol{\mu}_{\mathbf{x}_l},0.3^2)\text{ for } l=1,2, \\
&\quad \beta_i^t=3.5,\ \boldsymbol{\mu}_{\mathbf{x}}^{(0)}=[5.0,5.0]^T\ i=1,2,3.
\end{aligned}
\tag{3.15}
$$

It can be seen from Table 3.1 that SLShV-CG consumes more function evaluations than SLSV and SLSV-CG in order to generate a reliable optimum solution. This is because CG shows slow convergence for the convex function $G_1(\mathbf{X})$. PMA-CGA generates a better solution than SLSV and SLSV-CG but with the expense of g_{FC}.

It can be seen from β_{MCS}^t columns of Table 3.1 that SLShV-CG again emerges as the only method that can evolve the optimal solution with the desired target reliability. Other methods like SLSV and SLSV-CG are unable to generate reliable optimal solution. On the other hand, PMA-CGA is able to generate a solution that satisfies target reliability for constraint $G_2(\mathbf{X})$. PMA-AMV, SORA and ASORA are unable to converge for this example because the MPP is updated by steepest descent, which fails for concave functions.

The convergence of the solutions with respect to the number of iterations is shown in Figure 3.3. It can be seen that SLShV-CG requires more iterations in order to generate a reliable optimal solution. The progress of solutions obtained through SLShV-CG is shown in Figure 3.4, which indicates target reliability achieved by the optimal solution.

3.3.2 Example 2

The second mathematical example (Jeong and Park, 2017) as shown in equation (3.16) contains a non-linear objective function and a highly concave constraint. This example is considered to check the robustness of SLShV-CG for concave function. The initial point is taken as

Table 3.1: RBDO results for example 1 with $\beta^t = 3.5$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	NFC		β^t_{MCS}			Iter
			$\mathrm{f_{FC}}$	$\mathrm{g_{FC}}$	G_1	G_2	G_3	
DO	-2.292	$(5.197, 0.741)$	–	–	–	–	–	–
PMA-AMV	–	–	–	–	–	–	–	–
PMA-CGA	-1.6409	$(4.5273, 2.1587)$	51	16490	3.4808	3.6522	Inf	11
SLSV	-1.7289	$(4.9177, 1.9488)$	56	185	3.4808	2.3279	Inf	3
SLSV-CG	-1.6595	$(4.5965, 2.1131)$	92	290	3.4601	3.4227	Inf	4
SORA	–	–	–	–	–	–	–	–
ASORA	–	–	–	–	–	–	–	–
SLShV-CG (Proposed)	-1.6432	$(4.5225, 2.1537)$	198	699	3.5401	3.6949	Inf	13

'–' indicates that the method didnot converge.

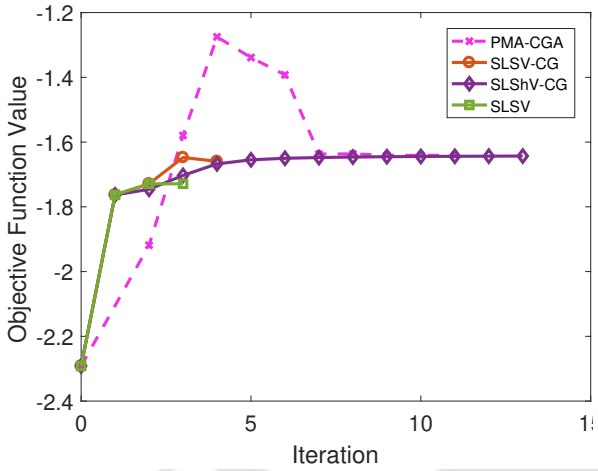


Figure 3.3: Convergence plot of example 1

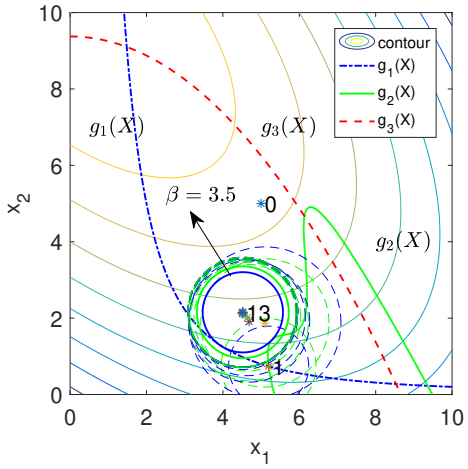


Figure 3.4: Contour plot of example 1

$\boldsymbol{\mu}^{(0)}_{\mathbf{x}} = [5.0, 5.0]^T$ and the standard deviation of the problem is $\boldsymbol{\sigma} = [0.6, 0.6]^T$. The lower and upper bounds of mean values of the random variables are 0.0 and 10.0, respectively. Table 3.2 presents the optimal solutions obtained from all methods.

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1}, \mu_{x_2}]^T \\
&\text{min.: } (\mu_{x_1} + 2)^2 + (\mu_{x_2} + 2)^2 - 2\mu_{x_1}\mu_{x_2}, \\
&\text{s.t.: } Pr\left[\frac{e^{(0.8x_1 - 1.2)} + e^{(0.7x_2 - 0.6)} - 5}{10} > 0\right] \leq \phi(-\beta^t_1), \qquad (3.16) \\
&\quad 0 \leq \boldsymbol{\mu}_{\mathbf{x}_l} \leq 10, \;\; x_l \sim N(\boldsymbol{\mu}_{\mathbf{x}_l}, 0.6^2) \text{ for } l = 1, 2, \\
&\quad \beta^t_1 = 3.0, \;\; \boldsymbol{\mu}^{(0)}_{\mathbf{x}} = [5.0, 5.0]^T.
\end{aligned}
$$

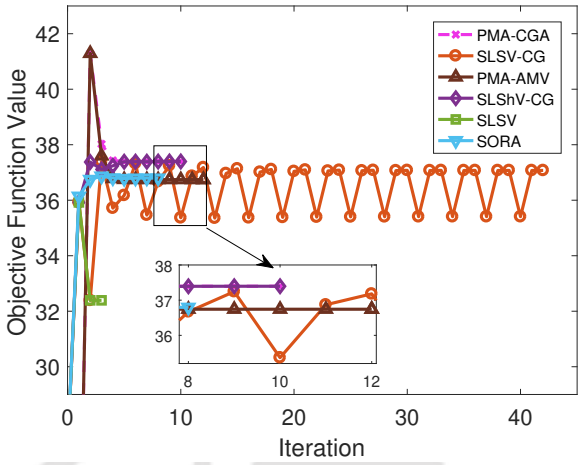


Figure 3.5: Convergence plot of example 2

It can be observed from Table 3.2 that SLShV-CG is found better than other methods, except for SLSV when comparing their NFC. However, SLSV has evolved the worst reliable solution among all methods. The convergence of all methods with respect to the number of iterations is shown in Figure 3.5. A smooth convergence is not observed with any method. It is due to the non-linear concave probabilistic constraint.

The accuracy of all these methods can be investigated by observing β^t_{MCS} columns of Table 3.2. It can be seen that SLShV-CG and PMA-CGA are the only methods which can evolve the solution with the desired target reliability. This is because PMA-CGA and SLShV-CG both utilize CG update to locate the MPP. On the other hand, PMA-AMV, SLSV, SORA and ASORA use steepest descent update for MPP calculation. Steepest descent works well for convex function but found unstable while handling concave function. Therefore, these methods shows poor convergence.

3.3.3 Example 3

The third example given in equation (3.17) is a non-linear mathematical problem (Yi et al., 2016), which has linear objective function and three non-linear constraints. Constraints $G_1(\mathbf{X})$ and $G_3(\mathbf{X})$ are convex functions and $G_2(\mathbf{X})$ is slightly concave in nature. Random variables x_1 and x_2 are normally distributed and statistically independent. Both these design variables have lower and upper bound set on their mean values to 0 and 10, respectively with standard deviation of 0.3. The initial point of this example is taken as, $\boldsymbol{\mu}_x^{(0)} = [5.0, 5.0]^T$ and the target

Table 3.2: RBDO results for example 2 with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	NFC		β^t_{MCS}	Iter
			$\mathrm{f_{FC}}$	$\mathrm{g_{FC}}$	G_1	
DO	27.0180	(3.1595, 3.8920)	–	–	–	–
PMA-AMV	36.7425	(3.1595, 3.8920)	53	10431	2.8538	12
PMA-CGA	37.3957	(3.5760, 3.7641)	33	993	3.0701	10
SLSV	32.3925	(1.8332, 3.5382)	116	127	1.3619	3
SLSV-CG	37.0741	(3.0822, 3.9833)	1248	1336	2.9517	42
SORA	36.7965	(2.9152, 3.9933)	190	2176	2.8109	8
ASORA	–	–	–	–	–	–
SLShV-CG (Proposed)	37.3956	(3.5715, 3.7677)	278	310	3.0729	10

reliability index β^t_i for all constraints is set to 3.0.

$$
\begin{aligned}
&\text{Find: } [\boldsymbol{\mu}_{x_1}, \boldsymbol{\mu}_{x_2}]^T \\
&\text{min: } \boldsymbol{\mu}_{x_1} + \boldsymbol{\mu}_{x_2}, \\
&\text{s.t.: } Pr\left[G_1(\mathbf{X}) = 1 - \frac{x_1^2 x_2}{20} > 0\right] \leq \phi(-\beta^t_1), \\
&\qquad Pr\left[G_2(\mathbf{X}) = 1 - \frac{(x_1 + x_2 - 5)^2}{30} - \frac{(x_1 - x_2 - 12)^2}{120} > 0\right] \\
&\qquad \leq \phi(-\beta^t_2), \\
&\qquad Pr\left[G_3(\mathbf{X}) = 1 - \frac{80}{(x_1^2 + 8x_2 + 5)} > 0\right] \leq \phi(-\beta^t_3), \\
&\qquad 0 \leq \boldsymbol{\mu}_{x_l} \leq 10,\ x_l \sim N(\boldsymbol{\mu}_{x_l}, 0.3^2) \text{ for } l = 1, 2, \\
&\qquad \beta^t_i = 3.0,\ \boldsymbol{\mu}^{(0)}_x = [5.0, 5.0]^T,\ i = 1, 2, 3.
\end{aligned}
\tag{3.17}
$$

Table 3.3 presents the obtained reliable solution by the methods.. It can be seen from the table that SLShV-CG is found to be computationally efficient than PMA-AMV when comparing their $\mathrm{f_{FC}}$ and $\mathrm{G_{FC}}$ in the fourth and the fifth columns of the table. Comparing the methods based on NFC, SLSV, SLSV-CG and ASORA are found to be efficient than SLShV-CG. However, SLShV-CG generates the reliable solution by compromising with NFC.

Regarding accuracy, it can be seen from β^t_{MCS} columns of Table 3.3 that only SLShV-CG and PMA-AMV methods evolve the optimal solutions with the desired target reliability. Other methods like DLM-CGA, SLSV-CG, SORA and ASORA are unable to achieve the target reliability for $G_1(\mathbf{X})$. It can also be seen that SLSV is unable to achieve the target reliability for both $G_1(\mathbf{X})$ and $G_2(\mathbf{X})$. It is noted that the constraint $G_3(\mathbf{X})$ is found to be inactive

Table 3.3: RBDO results for example 3 with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}_x^*$	NFC		β^t_{MCS}			Iter
			$\mathrm{f_{FC}}$	$\mathrm{g_{FC}}$	G_1	G_2	G_3	
DO	5.172	(3.111, 2.061)	-	-	-	-	-	-
PMA-AMV	6.7219	(3.4363, 3.2855)	27	5193	3.0045	3.0307	Inf	8
PMA-CGA	6.7256	(3.4391, 3.2865)	25	6157	2.9698	3.0068	Inf	8
SLSV	6.7199	(3.4419, 3.2779)	67	191	2.9402	2.9719	Inf	3
SLSV-CG	6.7253	(3.4393, 3.2861)	76	225	2.9576	3.0459	Inf	4
SORA	6.7226	(3.4369, 3.2857)	76	1137	2.9989	3.0617	Inf	4
ASORA	6.7256	(3.4391, 3.2865)	96	312	2.9698	3.0068	Inf	6
SLShV-CG (Proposed)	6.7224	(3.4371, 3.2853)	124	402	3.0258	3.0332	Inf	9

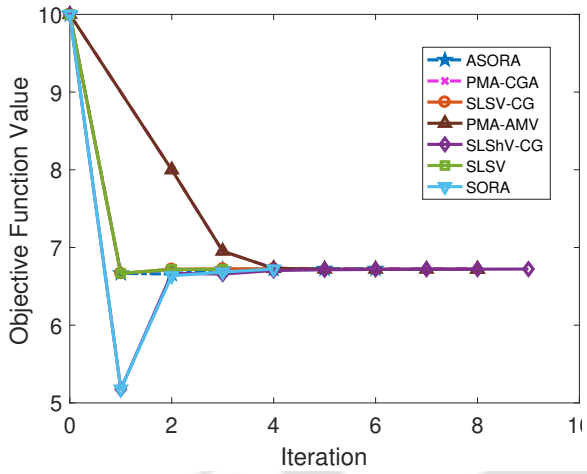


Figure 3.6: Convergence plot of example 3

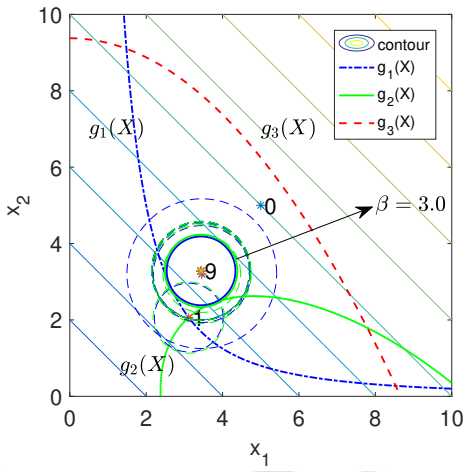


Figure 3.7: Contour plot of example 3

and thus, its target reliability is infinite for the solutions obtained by all methods. Figure 3.6 shows the convergence plots for all methods with respect to the number of iterations. The maximum number of iterations required to evolve the optimal solution is also mention in the last column of Table 3.3. A smooth convergence of SLShV-CG can be observed with respect to other methods.

The progress of the solution obtained through SLShV-CG in two-variable space is shown in Figure 3.7. The β circles for constraints $G_1(\mathbf{X})$ and $G_2(\mathbf{X})$ are shown, whereas for constraint $G_3(\mathbf{X})$ is not shown because the target reliability index is infinity. Graphically, it can be seen that SLShV-CG evolved the optimal solution with the desired target reliability.

3.3.4 Example 4

Another mathematical example shown in equation (3.18) is the problem no. 113 of Hock and Schittkowski (1981). The original problem is a deterministic problem, thus this RBDO example is adopted from Lee and Lee (2005). There are ten statistically independent random variables with normal distribution and eight probabilistic performance functions with the desired reliability index of 3.0. The optimal solution obtained from deterministic optimization is selected as the initial design point.

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1}, \mu_{x_2}, \mu_{x_3}, \mu_{x_4}, \mu_{x_5}, \mu_{x_6}, \mu_{x_7}, \mu_{x_8}, \mu_{x_9}, \mu_{x_{10}}]^T \\
&\text{min.: } \mu_{x_1}^2 + \mu_{x_2}^2 + \mu_{x_1}\mu_{x_2} - 14\mu_{x_1} - 16\mu_{x_2} + (\mu_{x_3} - 10)^2 \\
&\quad + 4(\mu_{x_4} - 5)^2 + (\mu_{x_5} - 3)^2 + 2(\mu_{x_6} - 1)^2 + 5\mu_{x_7}^2 \\
&\quad + 7(\mu_{x_8} - 11)^2 + 2(\mu_{x_9} - 10)^2 + (\mu_{x_{10}} - 7)^2 + 45 \\
&\text{s.t.: } Pr\left[G_j(\mathbf{X}) > 0\right] \le \phi(-\beta_i^t),\ \beta_i^t = 3.0,\ \text{for } i = 1, \ldots, 8, \\
&\quad G_1(\mathbf{X}) = \frac{4x_1 + 5x_2 - 3x_7 + 9x_8}{105} - 1 > 0, \\
&\quad G_2(\mathbf{X}) = 10x_1 - 8x_2 - 17x_7 + 2x_8 > 0, \\
&\quad G_3(\mathbf{X}) = \frac{-8x_1 + 2x_2 + 5x_9 - 2x_{10}}{12} - 1 > 0, \\
&\quad G_4(\mathbf{X}) = \frac{3(x_1 - 2)^2 + 4(x_2 - 3)^2 + 2x_3^2 - 7x_4}{120} - 1 > 0, \\
&\quad G_5(\mathbf{X}) = \frac{5x_1^2 + 8x_2 + (x_3 - 6)^2 - 2x_4}{40} - 1 > 0, \\
&\quad G_6(\mathbf{X}) = \frac{0.5(x_1 - 8)^2 + 2(x_2 - 4)^2 + 3x_5^2 - x_6}{30} - 1 > 0, \\
&\quad G_7(\mathbf{X}) = x_1^2 + 2(x_2 - 2)^2 - x_1x_2 + 14x_5 - 6x_6 > 0, \\
&\quad G_8(\mathbf{X}) = -3x_1 + 6x_2 + 12(x_9 - 8)^2 - 7x_{10} > 0, \\
&\quad 0 \le \boldsymbol{\mu}_{\mathbf{x}_l} \le 10,\ x_l \sim N(\boldsymbol{\mu}_{\mathbf{x}_l}, 0.02^2),\ \text{for } l = 1, 2, \ldots, 10, \\
&\quad \boldsymbol{\mu}_{\mathbf{x}}^{(0)} = [2.17, 2.36, 8.77, 5.10, 0.99, 1.43, 1.32, 9.83, 8.28, 8.38]^T.
\end{aligned}
\tag{3.18}
$$

Results obtained from all methods are summarized in Table 3.4. It can be observed that all the methods converge to a similar solution with an objective function value of 27.7466. The efficiency of all the methods is investigated by comparing their NFC. It can be seen that SLShV-CG has similar efficiency as compared with SLSV and SLSV-CG, and is better than other methods. It can also be observed that PMA-AMV and PMA-CGA require many NFC. This is because of the nested optimization, in which the inner loop calls the constraint function

several times at every iteration to calculate the gradient. However, SLShV-CG is a single-loop method thus requiring less number of constraint function calls.

Table 3.4: RBDO results for example 4 with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}_{\mathbf{x}}^*$	NFC		β_{MCS}^t	Iter
			f_{FC}	g_{FC}	G_i	
DO	24.3062	(2.1720, 2.3637, 8.7739, 5.0960, 0.9907, 1.4306, 1.3216, 9.8287, 8.2801, 8.3759)	–	–	–	–
PMA-AMV	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	184	666272	3.0280, 3.0045, 2.9781, 2.9824, 3.0000,inf, 3.0332, inf	14
PMA-CGA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	205	541500	3.0280, 3.0045, 2.9781, 2.9824, 3.0000,inf, 3.0332, inf	16
SLSV	27.7466	(2.1350, 2.3308, 8.7106, 5.1026, 0.9238, 1.4449, 1.3847, 9.8185, 8.1501, 8.4799)	287	1449	2.9957, 3.0332, 3.0013,, 3.0307, 3.0185, inf, 3.0407, inf	2
SLSV-CG	27.7466	(2.1350, 2.3308, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	289	1449	2.9957, 3.0332, 3.0013, 3.0307, 3.0185, inf, 3.0407, inf	2
SORA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	496	17031	3.0280, 3.0045, 2.9781, 2.9824, 3.0000,inf, 3.0332, inf	3
ASORA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	376	2159	2.9957, 3.0332, 3.0013,, 3.0307, 3.0185, inf, 3.0407, inf	2
SLShV-CG (Proposed)	27.7466	(2.1350, 2.3308, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	289	1439	2.9957, 3.0332, 3.0013, 3.0307, 3.0185, inf, 3.0407, inf	2

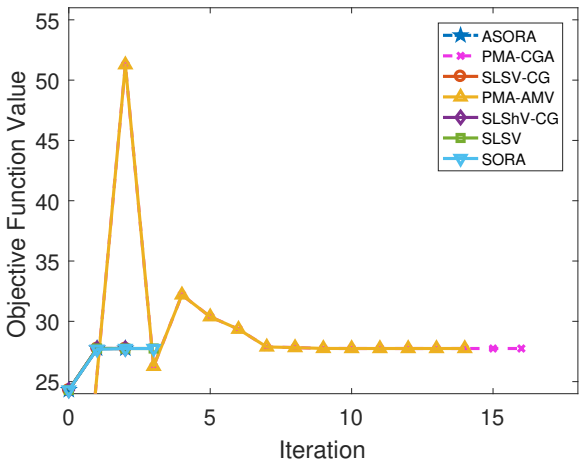


Figure 3.8: Convergence plot of example 4

The accuracy of these methods is investigated from β^t_{MCS} columns of Table 3.4. It can be seen that none of the methods are able to generate the optimal solution with the desired target reliability for all probabilistic constraints. SLShV-CG, SLSV, SLSV-CG, and ASORA evolve the same optimal solution, which is having the desired target reliability index closer to 3.0 for $G_1(\mathbf{X})$. Other methods like PMA-AMV, PMA-CGA and SORA are unable to achieve the desired target reliability for both $G_3(\mathbf{X})$ and $G_4(\mathbf{X})$. Figure 3.8 shows the convergence of all methods with respect to the number of iterations, which are found to be smooth for all methods.

3.3.5 Speed reducer example

A speed reducer example (Rao, 2009), as illustrated in Figure 3.9 is taken as the first engineering RBDO example. The design objective is to minimize the weight of the speed reducer, which is subjected to the constraints on bending stress, contact stress, longitudinal displacement stress, stress of the shaft, transverse deflection and geometric conditions. It contains seven independent random normal variables, each with standard deviation $\sigma = 0.005$ and 11 probabilistic performance functions with target reliability index of 3.0. The design variables are gear teeth (x_1), teeth module (x_2), number of teeth in pinion (x_3), the distance between two bearings (x_4, x_5) and axis diameter (x_6, x_7). The problem formulation is given in equation (3.19). Table 3.5 presents the optimal solutions obtained by the methods.

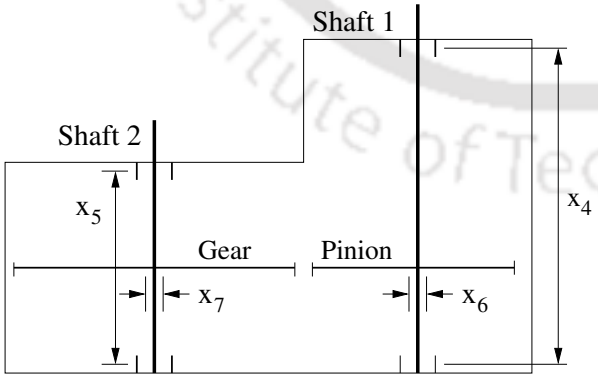


Figure 3.9: A speed reducer example

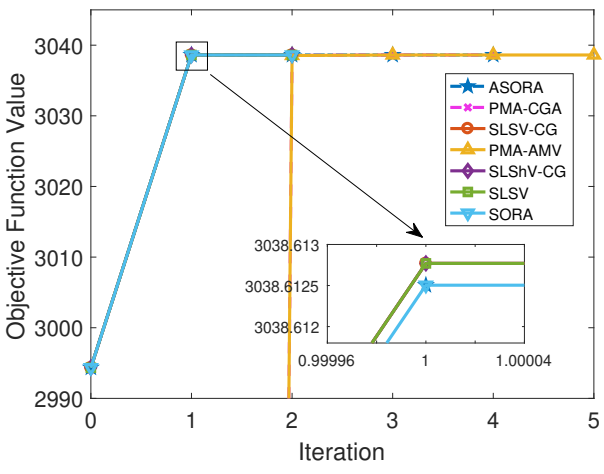


Figure 3.10: Convergence plot of speed reducer example

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1}, \mu_{x_2}, \mu_{x_3}, \mu_{x_4}, \mu_{x_5}, \mu_{x_6}, \mu_{x_7}]^T \\
&\text{min.: } 0.7854\mu_{x_1}\mu_{x_2}^2(3.3333\mu_{x_3}^2 + 14.9334\mu_{x_3} - 43.0934) - \\
&\quad 1.508\mu_{x_1}(\mu_{x_6}^2 + \mu_{x_7}^2) + 7.477(\mu_{x_6}^3 + \mu_{x_7}^3) \\
&\quad + 0.7854(\mu_{x_4}\mu_{x_6}^2 + \mu_{x_5}\mu_{x_7}^2), \\
&\text{s.t.: } Pr\left[G_i(\mathbf{X}) > 0\right] \leq \phi(-\beta_i^t), \\
&\quad G_1(\mathbf{X}) = \frac{27}{x_1 x_2^2 x_3} - 1 > 0,\ G_2(\mathbf{X}) = \frac{397.5}{x_1 x_2^2 x_3^2} - 1 > 0, \\
&\quad G_3(\mathbf{X}) = \frac{1.93x_4^3}{x_2 x_3 x_6^4} - 1 > 0,\ G_4(\mathbf{X}) = \frac{1.93x_5^3}{x_2 x_3 x_7^4} - 1 > 0, \\
&\quad G_5(\mathbf{X}) = \frac{\sqrt{(\frac{745x_4}{x_2 x_3})^2 + 16.9 \times 10^6}}{0.1x_6^3} - 1100 > 0, \\
&\quad G_6(\mathbf{X}) = \frac{\sqrt{(\frac{745x_5}{x_2 x_3})^2 + 157.5 \times 10^6}}{0.1x_7^3} - 850 > 0, \\
&\quad G_7(\mathbf{X}) = x_2 x_3 - 40 > 0,\ G_8(\mathbf{X}) = 5 - \frac{x_1}{x_2} > 0, \\
&\quad G_9(\mathbf{X}) = \frac{x_1}{x_2} - 12 > 0,\ G_{10}(\mathbf{X}) = \frac{1.5x_6 + 1.9}{x_4} - 1 > 0, \\
&\quad G_{11}(\mathbf{X}) = \frac{1.1x_7 + 1.9}{x_5} - 1 > 0, \\
&\quad 2.6 \leq x_1 \leq 3.6,\ 0.7 \leq x_2 \leq 0.8,\ 17 \leq x_3 \leq 28, \\
&\quad 7.3 \leq x_4 \leq 8.3,\ 7.3 \leq x_5 \leq 8.3,\ 2.9 \leq x_6 \leq 3.9, \\
&\quad 5 \leq x_7 \leq 5.5, \\
&\quad x_l \sim N(\boldsymbol{\mu}_{\mathbf{x}_l}, 0.005^2), \text{for } l = 1, 2, \ldots, 7, \\
&\quad \beta_i^t = 3.0,\ i = 1, 2, \ldots, 11, \\
&\quad \boldsymbol{\mu}_{\mathbf{x}}^{(0)} = [3.5, 0.7, 17, 7.3, 7.72, 3.35, 5.29]^T.
\end{aligned}
\tag{3.19}
$$

The efficiency of all the methods is investigated by comparing their NFC. SLShV-CG is found to be better than other methods. Among all, PMA-AMV seems to be the least efficient. On the other hand, PMA-CGA is twice efficient than PMA-AMV. Methods like SLSV and SLSV-CG show similar computational efficiency.

The accuracy is investigated from β_{MCS}^t columns of Table 3.5. It can be seen that performance functions $G_5(\mathbf{x}), G_6(\mathbf{x}), G_8(\mathbf{x})$ and $G_{11}(\mathbf{x})$ are active and rest of them are inactive at the optimal solution. For this example, all RBDO methods are converged to the same optimal

Table 3.5: RBDO results for speed reducer example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	NFC		β^t_{MCS}	Iter
			f_{FC}	g_{FC}	G_i	
PMA-AMV	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	229680	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	5
PMA-CGA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	120240	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	5
SLSV	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	1038	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	2
SLSV-CG	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	1038	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	2
SORA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	14874	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	3
ASORA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	112	1520	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	4
SLShV-CG (Proposed)	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	76	1014	Inf,Inf,Inf,Inf,3.0307, 3.3082, Inf,3.0000,Inf,Inf,3.0068	2

solution with the desired target reliability.

Figure 3.10 shows the convergence of all method with respect to the number of iterations. It can be noted that SLShV-CG requires only two iterations for convergence.

3.3.6 Tension/Compression spring example

A tension/compression spring (Meng and Keshtegar, 2018) is illustrated in Figure 3.11 and RBDO formulation is described in equation (3.20). The objective of the example is to minimize the spring weight. This example has three statistically independent random variables with normal distribution, namely mean coil diameter (x_1), the wire diameter (x_2) and the number of coils (x_3) and four probabilistic performance functions. The target reliability is fixed to 3.0 for all performance functions. The initial design point is selected as $\mu_x^{(0)} = [0.05, 0.5, 10]^T$. Table 3.6 summarizes the results obtained from all methods.

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1}, \mu_{x_2}, \mu_{x_3}]^T \\
&\text{min.: } (\mu_{x_3} + 2)\mu_{x_2}\mu_{x_1}^2, \\
&\text{s.t.: } Pr\left[G_i(\mathbf{X}) > 0\right] \leq \phi(-\beta_i^t),\ i = 1, 2, 3, 4, \\
&\qquad G_1(\mathbf{X}) = \frac{x_2^3 x_3}{71785 x_1^4} - 1 > 0, \\
&\qquad G_2(\mathbf{X}) = 1 - \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} - \frac{1}{5108 x_1^2} > 0,
\end{aligned}
\tag{3.20}
$$

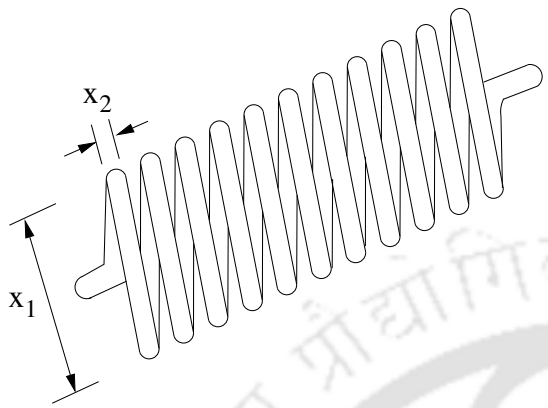


Figure 3.11: A spring tension/compression example

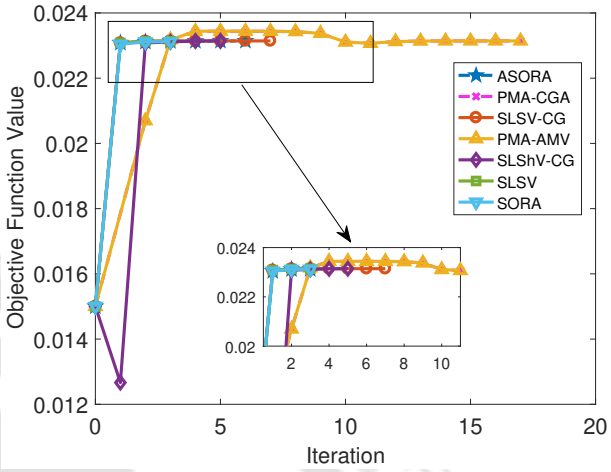


Figure 3.12: Convergence plot of spring tension/compression example

$$G_3(\mathbf{X}) = \frac{140.45x_1}{x_2^2 x_3} - 1 > 0,$$

$$G_4(\mathbf{X}) = 1 - \frac{x_1 + x_2}{1.5} > 0,$$

$$0.01 \leq \mu_{x_1} \leq 0.1,\ 0.1 \leq \mu_{x_2} \leq 1.0,\ 5.0 \leq \mu_{x_3} \leq 15.0,$$

$$x_1 \sim N(\boldsymbol{\mu}_{x_1}, 0.001^2),\ x_2 \sim N(\boldsymbol{\mu}_{x_2}, 0.01^2),$$

$$x_3 \sim N(\boldsymbol{\mu}_{x_3}, 0.8^2),\ \beta_i^t = 3.0,\ \boldsymbol{\mu}_x^{(0)} = [0.05, 0.5, 10]^T.$$

The efficiency of all the methods is investigated by comparing their NFC. SLShV-CG is found to be better than all methods, except for SLSV. However, SLSV is unable to generate a reliable solution. Methods like SLSV-CG and ASORA have comparable efficiency. PMA-AMV and PMA-CGA are the least efficient methods because of the double loop structure.

Regarding accuracy, it can be seen from β_{MCS}^t columns of Table 3.6 that only SLShV-CG converges to the optimal solution with desired target reliability and other methods fail to generate a reliable solution. Methods like SLSV and SORA converge to a solution with poor reliability as compared to other methods.

Figure 3.12 shows the convergence plot of all methods. It can be seen from the figure that SLShV-CG converges to the optimal solution within five iterations.

Table 3.6: RBDO results for spring tension/compression example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	NFC		β^t_{MCS}	Iter
			f_{FC}	g_{FC}	G_i	
PMA-AMV	0.02314298	(0.0589, 0.4620, 12.4319)	72	24480	2.9803, 3.0091 Inf,Inf	17
PMA-CGA	0.02314203	(0.0589, 0.4621,) 12.4286)	72	28744	2.9760, 3.0000 Inf,Inf	17
SLSV	0.02313374	(0.0586, 0.4545, 12.7981)	202	653	3.0000, 2.9576, Inf,Inf	3
SLSV-CG	0.02314299	(0.0589, 0.4607, 12.4970)	344	1281	2.9867, 3.0433, Inf,Inf	7
SORA	0.02312068	(0.0593, 0.4730, 11.9008)	210	1979	2.9781, 3.0091, Inf,Inf	3
ASORA	0.02314015	(0.0590, 0.4654, 12.2680)	321	1155	2.9824, 3.0045, Inf,Inf	6
SLShV-CG (Proposed)	0.02314287	(0.0590, 0.4649, 12.2908)	256	919	3.0068, 3.0282, Inf,Inf	5

3.3.7 Welded beam design example

A welded beam (Cho and Lee, 2011) as shown in Figure 3.13 is taken as the next engineering RBDO example. The objective function is to minimize the welding cost. There are five probabilistic performance functions related to the physical quantities such as shear stress, bending stress, bucking and tip deflection. Design variables such as depth (x_1) and length (x_2) of the welding, and height (x_3) and thickness (x_4) of the beam are statistically independent random variables with normal distributions. The RBDO model of the welded beam is given in equation (3.21). The fixed system parameters of equation (3.21) are listed in Table 3.7. Table 3.8 presents the results obtained from all methods.

$$
\begin{aligned}
&\text{Find: } [\mu_{x_1}, \mu_{x_2}, \mu_{x_3}, \mu_{x_4}]^T \\
&\text{min.: } c_1\mu_{x_1}^2\mu_{x_2} + c_2\mu_{x_3}\mu_{x_4}(z_2 + \mu_{x_2}), \\
&\text{s.t.: } P\left[G_1(\mathbf{X}) = \frac{\tau(\mathbf{X},\mathbf{z})}{z_6} - 1 > 0\right] \leq \phi(-\beta_1^t), \\
&\quad P\left[G_2(\mathbf{X}) = \frac{\sigma(\mathbf{X},\mathbf{z})}{z_7} - 1 > 0\right] \leq \phi(-\beta_2^t), \\
&\quad P\left[G_3(\mathbf{X}) = \frac{x_1}{x_4} - 1 > 0\right] \leq \phi(-\beta_3^t), \\
&\quad P\left[G_4(\mathbf{X}) = \frac{\delta(\mathbf{X},\mathbf{z})}{z_5} - 1 > 0\right] \leq \phi(-\beta_4^t), \\
&\quad P\left[G_5(\mathbf{X}) = 1 - \frac{P_c(\mathbf{X},\mathbf{z})}{z_1} > 0\right] \leq \phi(-\beta_5^t), \\
&\quad 3.175 \leq x_1 \leq 50.8,\ 0 \leq x_2 \leq 254,\ 0 \leq x_3 \leq 254, \\
&\quad 0 \leq x_4 \leq 50.8, \\
&\quad x_{1,2} \sim N(\boldsymbol{\mu}_{x_{1,2}}, 0.1693^2),\ x_{3,4} \sim N(\boldsymbol{\mu}_{x_{3,4}}, 0.0107^2), \\
&\quad \beta_i^t = 3.0,\ i = 1, 2, \ldots, 5, \\
&\quad \boldsymbol{\mu}_{\mathbf{x}}^{(0)} = [6.208, 157.82, 210.62, 6.208]^T, \\
&\quad t(\mathbf{X},\mathbf{z}) = \frac{z_1}{\sqrt{2}x_1x_2},\ tt(\mathbf{X},\mathbf{z}) = M(\mathbf{X},\mathbf{z})\frac{R(\mathbf{X},\mathbf{z})}{J(\mathbf{X},\mathbf{z})}, \\
&\quad M(\mathbf{X},\mathbf{z}) = z_1\left(z_2 + \frac{x_2}{2}\right),\ R(\mathbf{X},\mathbf{z}) = \frac{\sqrt{x_2^2 + (x_1 + x_3)^2}}{2}, \\
&\quad J(\mathbf{X},\mathbf{z}) = \sqrt{2}x_1x_2\left\{\frac{x_2^2}{12} + \frac{(x_1 + x_3)^2}{4}\right\}, \\
&\quad \sigma(\mathbf{X},\mathbf{z}) = \frac{6z_1z_2}{x_3^2x_4},\ \delta(\mathbf{X},\mathbf{z}) = \frac{4z_1z_2^3}{z_3x_3^3x_4}, \\
&\quad P_c(\mathbf{X},\mathbf{z}) = \frac{4.013x_3x_4^3\sqrt{z_3z_4}}{6z_2^2}\left(1 - \frac{x_3}{4z_2}\sqrt{\frac{z_3}{z_4}}\right), \\
&\quad \tau(\mathbf{X},\mathbf{z}) = \left\{t(\mathbf{X},\mathbf{z})^2 + 2t(\mathbf{X},\mathbf{z})tt(\mathbf{X},\mathbf{z})\left(\frac{x_2}{2R(\mathbf{X},\mathbf{z})}\right) + tt(\mathbf{X},\mathbf{z})^2\right\}^{1/2}.
\end{aligned}
\tag{3.21}
$$

Regarding efficiency, it can be seen from the fifth column of the table that among all the methods, SLShV-CG is the most efficient method. Other methods like SLSV, SLSV-CG and ASORA show better efficiency than PMA-AMV and PMA-CGA.

Comparing the accuracy for this engineering RBDO example from β_{MCS}^t columns of Table

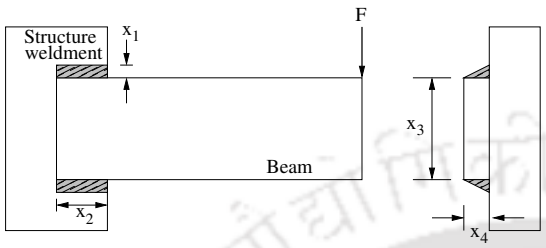


Figure 3.13: A welded beam structure example

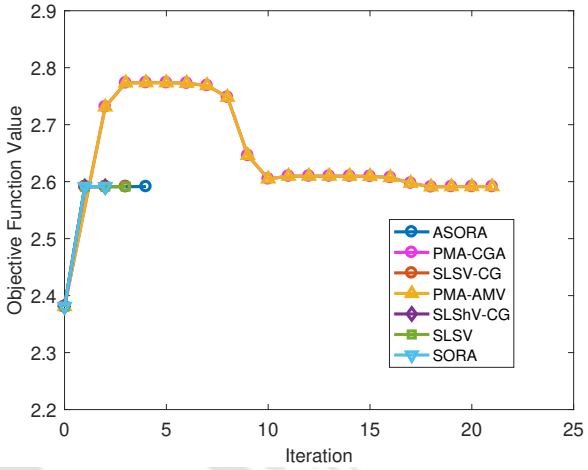


Figure 3.14: Convergence plot of welded beam example

Table 3.7: Fixed parameters for the welded beam example

z_1 :	2.6688×10^4 (N)
z_2 :	3.556×10^2 (mm)
z_3 :	2.0685×10^5 (MPa)
z_4 :	8.274×10^4 (MPa)
z_5 :	6.35 (mm)
z_6 :	9.377×10 (MPa)
z_7 :	2.0685×10^2 (MPa)
c_1 :	6.74135×10^{-5} (\$/mm^3)
c_2 :	2.93585×10^{-6} (\$/mm^3)

3.8, all methods converge to the same optimal solution with the desired target reliability. It can be noted from the sixth column of the table that $G_4(\mathbf{X})$ is an inactive performance function.

The convergence plot is shown in Figure 3.14, which indicates a smooth convergence of all methods, except PMA-AMV and PMA-CGA.

Table 3.8: RBDO results for welded beam design example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	NFC		β^t_{MCS}	Iter
			$\mathrm{f_{FC}}$	$\mathrm{g_{FC}}$	G_i	
PMA-AMV	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	115	57750	3.0233, 3.0233 3.0091,Inf,3.0091	21
PMA-CGA	2.5913	(5.7300, 200.8981, 210.5977, 6.2389)	110	37290	3.0233, 3.0233 3.0091,Inf,3.0091	21
SLSV	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	177	847	3.0233, 3.0233 3.0091,Inf,3.0091	3
SLSV-CG	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	177	847	3.0233, 3.0233 3.0091,Inf,3.0091	3
SORA	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	160	2155	3.0233, 3.0233 3.0091,Inf,3.0091	3
ASORA	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	193	905	3.0233, 3.0233 3.0091,Inf,3.0091	4
SLShV-CG (Proposed)	2.5913	(5.7300, 200.8982, 210.5977, 6.2389)	164	740	3.0233, 3.0233 3.0091,Inf,3.0091	3

3.3.8 Cantilever beam example

$$
\begin{aligned}
&\text{Find: } \mu_w,\ \mu_t \\
&\text{min: } w \times t, \\
&\text{s.t.: } P\left[G_i(\mathbf{X}) > 0\right] \le \phi(-\beta_i^t),\ i = 1, 2, \\
&\quad G_1(S_y, p_y, p_z, t) = S_y - \left(\frac{600}{wt^2}p_y + \frac{600}{wt^2}p_z\right) > 0, \\
&\quad G_2(E, w, t, p_y, p_z) = D_0 - \frac{4L^3}{Ewt}\sqrt{\left(\frac{p_y}{t^2}\right)^2 + \left(\frac{p_z}{w^2}\right)^2} > 0, \\
&\quad 0 \le w \le 5,\ 0 \le t \le 5,\ 500 \le p_z \le 800, \\
&\quad 1000 \le p_y \le 1500,\ 35000 \le S_y \le 45000, \\
&\quad 25 \times 10^6 \le E \le 30 \times 10^6, \\
&\quad w \sim N(\boldsymbol{\mu}_w, 0.01^2),\ t \sim N(\boldsymbol{\mu}_t, 0.01^2), \\
&\quad p_z \sim N(\boldsymbol{\mu}_{p_z}, 100^2), \\
&\quad p_y \sim N(\boldsymbol{\mu}_{p_y}, 100^2),\ S_y \sim N(\boldsymbol{\mu}_{S_y}, 2000^2), \\
&\quad E \sim N(\boldsymbol{\mu}_E, (1.45 \times 10^6)^2), \\
&\quad \beta_i^t = 3.0,\ \ \boldsymbol{\mu}^{(0)} = [2, 2, 500, 1000, 40000, 29 \times 10^6]^T\ .
\end{aligned}
\tag{3.22}
$$

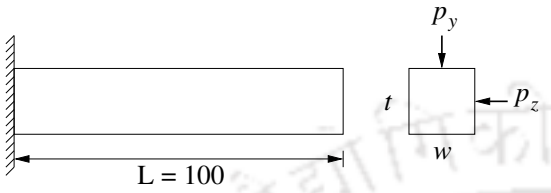


Figure 3.15: A cantilever beam example

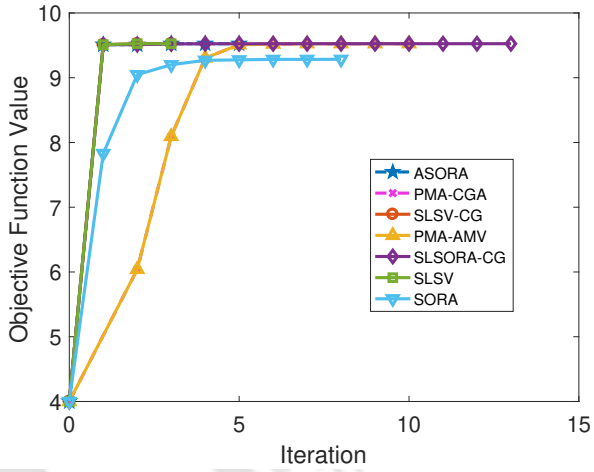


Figure 3.16: Convergence plot of cantilever beam example

A cantilever beam example (Li et al., 2013) adopted here is shown in Figure 3.15. The problem formulation is given in equation (3.22). The beam is loaded with two point loads, lateral load p_z and vertical load p_y at the tip. The width and thickness of the beam are 'w' and 't', respectively and the length L of the beam is equal to 100 inches. The objective of the example is to minimize the weight of the beam. The first performance function represents that the maximum stress at the fixed end that should be less than the yield strength S_y. The second performance function represents the displacement that should not exceed the allowable value of D_0. The desired reliability index of $\beta_i^t = 3.0$ is kept for both the performance functions.

The results for this example are summarized in Table 3.9. It can be seen from the table that SLShV-CG is found better than PMA-AMV, PMA-CGA and SORA when comparing their NFC. However, SLShV-CG needs improvement over SLSV, SLSV-CG and ASORA, which are computationally efficient.

Regarding accuracy, all methods evolve the same reliable solution for this example, except SORA. The convergence plot for all methods is shown in Figure 3.16. It can be seen that SLShV-CG requires more iterations than SLSV, SLSV-CG, ASORA and SORA.

3.4 Discussion

An accurate and computationally efficient hybrid RBDO method has been developed by coupling the single-loop method with the shifting vector approach of SORA. The approximate

Table 3.9: RBDO results for cantilever beam example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*$	NFE		β^t_{MCS}		Iter
			f_{FE}	g_{FE}	g_1	g_2	
PMA-AMV	9.5253	(2.4538, 3.8819)	77	13930	3.0025,	3.0320	10
PMA-CGA	9.5253	(2.4531, 3.8829)	77	26936	3.0357,	3.0045	10
SLSV	9.5253	(2.4538, 3.8819)	175	370	3.0025,	3.0320	3
SLSV-CG	9.5253	(2.4538, 3.8819)	182	384	3.0025,	3.0320	3
SORA	9.2840	(2.5802, 3.5981)	325	2498	2.5696,	3.0000	3
ASORA	9.5253	(2.4538, 3.8819)	260	570	3.0022,	3.0357	5
SLShV-CG (Proposed)	9.5252	(2.4538, 3.8819)	443	1122	3.0185,	3.0433	13

MPTP was updated using the conjugate gradient search direction in each iteration. Four mathematical and four engineering RBDO examples were solved for the verification of SLShV-CG. Examples 2 and 3 demonstrated that SLShV-CG converged to the desired reliable solution irrespective of the convex or concave nature of the performance functions, whereas other prominent methods failed to converge. Also, SLShV-CG is found to be computationally efficient than those methods, which are able to generate the reliable optimal solution.

As the proposed method utilizes CG for obtaining MPTP, there is always a chance of oscillation during the convergence of MPTP in case of highly non-linear performance function. Example 2 demonstrated such oscillation while converging to the reliable optimal solution. This leads to the second objective of this thesis in which these oscillations are traced and appropriate theory is applied. In the next section, an approximate single-loop chaos control method is discussed in detail.

3.5 Proposed approximate single-loop chaos control (ASLCC) method

The solution obtained by SLA may get stuck to periodic oscillation due to the use of steepest descent search or conjugate gradient search directions while estimating MPTPs for highly nonlinear performance functions. Therefore, the chaos control theory is used to obtain MPTP when it starts oscillating in the search space. In order to have proper understanding of the method a brief discussion of chaos control theory is presented.

3.5.1 Chaos control theory (CC)

Pingel et al. (2004) introduced an appropriate linear transformation to modify the eigenvalues of Jacobian matrix of dynamical system and stabilize the unstable fixed points of original system without changing their value and location. The iterative point $\boldsymbol{x}^{(k+1)}$ is updated according to equation (3.23).

$$\boldsymbol{x}^{(k+1)} = \boldsymbol{x}^{(k)} + \lambda \mathbf{C}[f(\boldsymbol{x}^{(k)}) - \boldsymbol{x}^{(k)}], \tag{3.23}$$

where λ is the chaos control factor which lies between 0 and 1. $\mathbf{C}$ is a $n \times n$ dimensional involutory matrix, i.e., a orthogonal matrix in which each row and each column will comprise of only one element 1 or -1 and other elements are 0. The total number of involutory matrix can be $2^n n!$. To enhance the efficiency of chaos control method, matrix $\mathbf{C}$ must be the minimum set of involutory matrix (i.e., minimum number of $2^n n!$ matrix). The selection of $\mathbf{C}$ also depends on the property of saddle point and spiral point of unstable fixed points. Generally, selecting the unit matrix of $\mathbf{C}$ (Pingel et al., 2004) can stabilize the unstable fixed points. Therefore, for a two dimensional dynamical system the matrix $\mathbf{C}$ can be taken as an identity matrix of size 2×2. The factor λ should be small enough to make the unstable fixed point stable. The selection of factor λ also depends on the original Jacobian matrix. The larger the maximum of the absolute eigenvalues of the Jacobian matrix, the smaller λ value should be taken for stabilization. This leads to more number of iteration in order to achieve stabilization. Chaos control factor also acts as an important parameter which controls the efficiency as well as accuracy of any RBDO method in which chaos control method performs the reliability analysis. It has also been observed that this chaos control factor is a sensitive parameter for convergence. Therefore, the efficiency of the RBDO methods becomes sensitive to the initial value of chaos control factor.

As mentioned earlier, the iterative points to update MPTP in PMA can be obtained by steepest descent direction or by conjugate gradient direction. As these directions can yields to periodic oscillation, chaos control method (Yang and Yi, 2009) is used to evaluate the iterative points of the MPTP search, which is calculated as

$$\mathbf{u}^{(k+1)} = \mathbf{u}^{(k)} + \lambda \mathbf{C}[f(\mathbf{u}^{(k)}) - \mathbf{u}^{(k)}], \tag{3.24}$$

where

$$f(\mathbf{u}^{(k)}) = \mathbf{u}_{AMV}^{(k)} = -\beta^t \frac{\nabla_{\mathbf{u}} G(\mathbf{u}_{AMV}^{(k)})}{\|\nabla_{\mathbf{u}} G(\mathbf{u}_{AMV}^{(k)})\|}, \tag{3.25}$$

In equation (3.24), the involutory matrix $\mathbf{C}$ is taken as Identity matrix ($\boldsymbol{I}$) and the value of chaos control factor λ is taken between 0 and 1. Equation (3.24) is used in place of equation (2.10) to update the MPTP in PMA.

The use of chaos control theory in the double-loop method makes it inefficient as it is implemented at every iteration. To have a better implementation of chaos control theory, a proper oscillation criterion for tracking oscillation during the convergence of MPTPs is needed. Therefore, a new RBDO method using the single-loop method with the chaos control theory is proposed to address these issues. In the following section, the proposed method is discussed in detail.

3.5.2 The proposed ASLCC method

As discussed earlier, SLA (Liang et al., 2008) employs KKT optimality conditions to approximate MPTP from the standard normal space to the normal space. This enhances the efficiency of RBDO method by eliminating the reliability analysis loop. The deterministic optimization that is used in RBDO formulation is as follows.

$$\begin{aligned} &\text{min: } f(\boldsymbol{\mu}_{\mathbf{X}}), \\ &\text{s.t.: } G_i(\mathbf{d}, \mathbf{X}_{i,MPTP}^{(k)}) \leq 0, \quad i = 1, \ldots, M, \end{aligned} \tag{3.26}$$

where $f(\boldsymbol{\mu}_{\mathbf{X}})$ is the objective function, G_i is the $i-$th deterministic performance function, $\mathbf{d}$ is the deterministic design variable and $\boldsymbol{\mu}_{\mathbf{X}}$ is the mean values of random variable vector $\mathbf{X}$. The approximated MPTP is expressed as

$$\mathbf{X}_{i,MPTP}^{(k)} = \mathrm{T}^{-1}(\mathbf{U}) = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} + \boldsymbol{\sigma}_{\mathbf{X}} \mathbf{U}_{i,ASLCC}^{(k)}, \tag{3.27}$$

where $\mathbf{U}$ is the vector of random variables in the standard normal space, $\boldsymbol{\sigma}_{\mathbf{X}}$ is the standard deviation of the random variables, $\mathbf{X}_{i,MPTP}^{(k)}$ is the approximated MPTP for $i-$th constraint in the normal space at $k-$th iteration. Since the accuracy of the reliable optimal solution depends on $\mathbf{X}_{i,MPTP}^{(k)}$, this thesis thus focuses on evaluating accurate $\mathbf{U}_{i,ASLCC}^{(k)}$ given in equation (3.27) using the proposed method. An oscillation criterion is also proposed to track MPTPs. When these MPTPs start oscillating, the current MPTP of equation (3.27) is updated using chaos control theory. The oscillation criterion is given in equation (3.28).

$$\gamma_i^{(k+1)} = (\mathbf{U}_{i,ASLCC}^{(k+1)} - \mathbf{U}_{i,ASLCC}^{(k)}) \cdot (\mathbf{U}_{i,ASLCC}^{(k)} - \mathbf{U}_{i,ASLCC}^{(k-1)})$$
$$\gamma_i^{(k+1)} > 0 : \text{no oscillation or converging of MPTPs} \tag{3.28}$$
$$\leq 0 : \text{oscillation or diverging of MPTPs}$$

where the value of $\gamma_i^{(k+1)}$ decides the oscillation among consecutive MPTPs of $i-$th performance function. If the dot product of equation (3.28) is positive, the two vectors $(\mathbf{U}_{i,ASLCC}^{(k+1)} - \mathbf{U}_{i,ASLCC}^{(k)})$ and $(\mathbf{U}_{i,ASLCC}^{(k)} - \mathbf{U}_{i,ASLCC}^{(k-1)})$ are almost in the same direction, signifying there is convergence as shown in Figure 3.18. In this case, the conjugate gradient search direction is used to estimate MPTP because it is found to be robust (Jeong and Park, 2017). On the other hand, if the dot product is negative, the two vectors $(\mathbf{U}_{i,ASLCC}^{(k+1)} - \mathbf{U}_{i,ASLCC}^{(k)})$ and $(\mathbf{U}_{i,ASLCC}^{(k)} - \mathbf{U}_{i,ASLCC}^{(k-1)})$ are moving away from each other. The angle between these vectors thus increases, signifying oscillation among MPTPs as shown in Figure 3.19. In this case, the chaos control theory is utilized to estimate the new MPTP, which helps to eliminate oscillation of MPTPs for highly nonlinear performance functions. Moreover, the control factor λ is determined from the angle updating criterion, which is given in equation (3.29).

$$\lambda = \begin{cases} 0.2\lambda, & 0.2\theta^{(k)} > \theta^{(k-1)}, \\ \lambda\frac{\theta^{(k-1)}}{\theta^{(k)}}, & \theta^{(k)} > \theta^{(k-1)} \geq 0.2\theta^{(k)}, \\ \lambda, & \theta^{(k)} \leq \theta^{(k-1)}, \end{cases} \tag{3.29}$$

where $\theta^{(k)}$ and $\theta^{(k-1)}$ are the angles between $(\mathbf{U}^{(k+1)},\mathbf{U}^{(k)})$ and $(\mathbf{U}^{(k)},\mathbf{U}^{(k-1)})$, respectively.

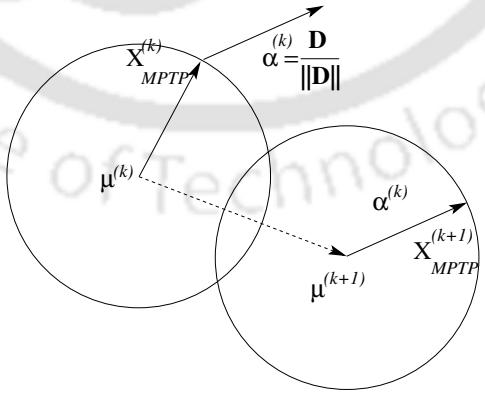


Figure 3.17: Schematic diagram for MPTP estimation using conjugate gradient method

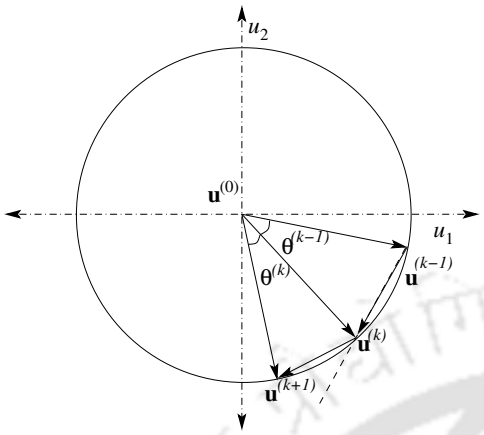


Figure 3.18: converging MPTP

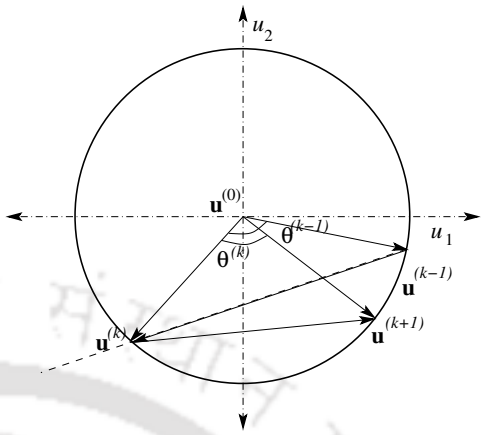


Figure 3.19: diverging MPTP

3.5.2.1 Estimation of MPTP using conjugate gradient search direction

The formulation to approximate MPTP is as follows

$$\mathbf{X}_{i,MPTP}^{(k)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} + \beta_i^t \boldsymbol{\sigma}_{\mathbf{X}} \boldsymbol{\alpha}_i^{(k-1)}, \tag{3.30}$$

$$\boldsymbol{\alpha}_i^{(k)} = \begin{cases} \dfrac{\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i}{\|\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i\|}\Big|_{\boldsymbol{\mu}_{\mathbf{X}}^{(0)}}, & \text{if } k = 0, \\ \dfrac{\mathbf{D}_i}{\|\mathbf{D}_i\|}\Big|_{\mathbf{X}_{i,MPTP}^{(k)}}, & \text{otherwise,} \end{cases} \tag{3.31}$$

where $\boldsymbol{\alpha}_i^{(k)}$ is the normalized direction at the $k-$th iteration that is estimated by the conjugate gradient search direction, $\mathbf{D}_i^{(k)}$. It is given as

$$\mathbf{D}_i^{(k)} = \begin{cases} \boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i, & k \leq 1 \\ \boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)} + \dfrac{(\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)})^T (\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)})}{(\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k-1)})^T (\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k-1)})} \cdot \mathbf{D}_i^{(k-1)}, & \text{otherwise,} \end{cases} \tag{3.32}$$

Figure 3.17 shows the iterative updating of MPTP using conjugate gradient search directions.

3.5.2.2 Estimation of MPTP using chaos control theory

After calculating $\mathbf{X}_{i,MPTP}^{(k)}$ using equation (3.30), it is transformed to the standard normal space to find the corresponding value of $\mathbf{U}_{i,MPTP}^{(k)}$ using equation (3.33). Thereafter, $\mathbf{U}_{i,ASLCC}^{(k)}$

is updated using equation (3.34).

$$\mathbf{U}_{i,MPTP}^{(k)} = \mathrm{T}(\mathbf{X}_i) = (\mathbf{X}_{i,MPTP}^{(k)} - \boldsymbol{\mu}_{\mathbf{X}})/\boldsymbol{\sigma}_X. \tag{3.33}$$

$$\mathbf{U}_{i,ASLCC}^{(k)} = \begin{cases} \mathbf{U}_{i,MPTP}^{(k)}, & \text{if } k < 2 \text{ or } \gamma_i > 0, \\ \beta_i^t \frac{\mathbf{U}_i^{(k-1)} + \lambda_i^{(k)} \mathbf{C}[\mathbf{U}_{i,MPTP}^{(k)} - \mathbf{U}_i^{(k-1)}]}{\|\mathbf{U}_i^{(k-1)} + \lambda_i^{(k)} \mathbf{C}[\mathbf{U}_{i,MPTP}^{(k)} - \mathbf{U}_i^{(k-1)}]\|}, & \text{otherwise.} \end{cases} \tag{3.34}$$

It can be seen that $\mathbf{U}_{i,ASLCC}^{(k)}$ is the same as $\mathbf{U}_{i,MPTP}^{(k)}$ when $k < 2$ or $\gamma_i > 0$ of equation (3.28). Otherwise, $\mathbf{U}_{i,ASLCC}^{(k)}$ is evaluated separately using chaos control theory. The chaos control factor $\lambda_i^{(k)}$ is changed in each iteration using the angle condition given in equation (3.29). If the angle ($\theta^{(k)}$) as shown in Figure 3.18 is increasing from the previous angle ($\theta^{(k-1)}$), $\lambda_i^{(k)}$ is decreased to $0.2 \times \lambda_i^{(k)}$, otherwise $\lambda_i^{(k)}$ is kept constant. The converging and diverging scenarios of MPTP in the standard normal space are shown in the Figure 3.18 and Figure 3.19, respectively. It can also be seen in Figure 3.18 that the consecutive MPTPs are in the same direction and therefore, achieve stable convergence. On the other hand, if the direction of vectors changes, it signifies divergence or oscillation as shown in Figure 3.19.

The proposed method is referred to as approximate single-loop chaos control method (ASLCC) because the chaos control method is used when MPTPs start oscillating. The following are the steps for ASLCC method and the flowchart is shown in Figure 3.20, in which all steps are shown.

Step 1. Initialize with mean values of random variables, $\boldsymbol{\mu}_{\mathbf{X}}^{(0)}$, standard deviation of the random variables, $\boldsymbol{\sigma}_{\mathbf{X}}$, target reliability index of the performance function, β^T, chaos control factor, $\lambda_i^{(k)}$, and the involutory matrix, $\mathbf{C}$, that is considered as identity matrix $\mathbf{I}$.

Step 2. Set $k = 0$, initial design variables $\mathbf{d}^{(0)} = \boldsymbol{\mu}_{\mathbf{X}}^{(0)}$, initial MPTP as $\mathbf{X}_{MPTP}^{(0)} = \boldsymbol{\mu}_{\mathbf{X}}^{(0)}$ and $\mathbf{U}_{MPTP}^{(0)} = T(\mathbf{X}_{MPTP}^{(0)})$.

Step 3. Perform deterministic optimization as

$$\begin{aligned} &\min \quad f(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{X}}), \\ &\text{s.t.} \quad G_i(\mathbf{d}, \mathbf{X}_i^{(k)}) \le 0 \qquad\qquad i = 1, \ldots, M, \\ &\text{where} \quad \mathbf{X}_i^{(k)} = \begin{cases} \mathbf{X}_{MPTP}^{(0)}, & k = 0 \\ \boldsymbol{\mu}_{\mathbf{X}}^{(k)} + \boldsymbol{\sigma}_{\mathbf{X}}^T \mathbf{U}_{i,ASLCC}^{(k)}, & \text{otherwise,} \end{cases} \end{aligned} \tag{3.35}$$

and generate the mean value of random variables $\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}$.

Step 4. Calculate $\mathbf{D}_i^{(k)}$ using equation (3.36)

$$\begin{aligned}\mathbf{D}_i^{(k)} &= \begin{cases} \boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i, & k \leq 1 \\ \boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)} + \frac{(\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)})^T (\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k)})}{(\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k-1)})^T (\boldsymbol{\sigma}_{\mathbf{X}} \nabla G_i^{(k-1)})} \cdot \mathbf{D}_i^{(k-1)}, & \text{otherwise.} \end{cases} \\ \boldsymbol{\alpha}_i^{(k)} &= \frac{\mathbf{D}_i}{\|\mathbf{D}_i\|}\Big|_{\mathbf{X}_{i,MPTP}^{(k)}}, \end{aligned} \tag{3.36}$$

where, $\boldsymbol{\alpha}_i^{(k)}$ is estimated with conjugate gradient search direction $\mathbf{D}_i^{(k)}$ if $k \geq 2$. Otherwise, $\boldsymbol{\alpha}_i^{(k)}$ is estimated with steepest descent search direction,

$$\boldsymbol{\alpha}_i^{(k)} = \frac{\boldsymbol{\sigma}_{\mathbf{X}} \ \nabla G_{i_{(\mathbf{X})}}(\mathbf{d}^{(k)}, \mathbf{X}_i^{(k)})}{\|\boldsymbol{\sigma}_{\mathbf{X}} \ \nabla G_{i_{(\mathbf{X})}}(\mathbf{d}^{(k)}, \mathbf{X}_i^{(k)})\|}. \tag{3.37}$$

Step 5. Update $\mathbf{X}_{i,MPTP}^{(k+1)}$ in the normal space as

$$\mathbf{X}_{i,MPTP}^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k+1)} + \beta_i^t \boldsymbol{\sigma}_{\mathbf{X}} \boldsymbol{\alpha}_i^{(k)}, \tag{3.38}$$

and calculate $\mathbf{U}_{i,MPTP}^{(k+1)} = T(\mathbf{X}_{i,MPTP}^{(k+1)})$ and proceed.

Step 6. Check the value of oscillation criterion $\gamma^{(k)}$ when $k > 1$. If the value is positive or $k \leq 1$, estimate $\mathbf{U}_{i,ASLCC}^{(k+1)}$ as $\mathbf{U}_{i,MPTP}^{(k+1)}$ and goto step 8. Else, proceed to step 7.

Step 7. Assign $\mathbf{U}_{i,ASLCC}^{(k+1)}$ as given in equation (3.39) and proceed to Step 8.

$$\mathbf{U}_{i,ASLCC}^{(k+1)} = \beta^T \frac{\mathbf{U}_i^{(k)} + \lambda_i^{(k)} \mathbf{C}[\mathbf{U}_{i,MPTP}^{(k+1)} - \mathbf{U}_i^{(k)}]}{\|\mathbf{U}_i^{(k)} + \lambda_i^{(k)} \mathbf{C}[\mathbf{U}_{i,MPTP}^{(k+1)} - \mathbf{U}_i^{(k)}]\|} \tag{3.39}$$

Step 8. If the convergence criterion $\|f(\mathbf{d}^{(k+1)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k+1)}) - f(\mathbf{d}^{(k)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| / \|f(\mathbf{d}^{(k)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| \leq 0.001$ or $\|\boldsymbol{\mu}_{\mathbf{x}}^{(k+1)} - \boldsymbol{\mu}_{\mathbf{x}}^{(k)}\| \leq 0.001$ is satisfied, terminate. Otherwise, go to Step 9.

Step 9. Evaluate the angle between two vectors as

$$\theta^{(k)} = cos^{-1}\left(\frac{\mathbf{U}_{i,ASLCC}^{(k+1)} \cdot \mathbf{U}_{i,MPTP}^{(k)}}{\|\mathbf{U}_{i,ASLCC}^{(k+1)}\| \|\mathbf{U}_{i,MPTP}^{(k)}\|}\right), \tag{3.40}$$

and calculate the iterative chaos control factor for $k \geq 1$ as

$$\lambda_i^{(k+1)} = \begin{cases} 0.2\lambda_i^{(k)}, & 0.2\theta^{(k)} > \theta^{(k-1)}, \\ \lambda_i^{(k)} \frac{\theta^{(k-1)}}{\theta^{(k)}}, & \theta^{(k)} > \theta^{(k-1)} \geq 0.2\theta^{(k)}, \\ \lambda_i^{(k)}, & \theta^{(k)} \leq \theta^{(k-1)} \end{cases} \tag{3.41}$$

Step 10. Check the iteration counter k. If $k > 2$, calculate the oscillation criterion using equation (3.42). Else, goto step 11.

$$\gamma_i^{(k+1)} = (\mathbf{U}_{i,ASLCC}^{(k+1)} - \mathbf{U}_{i,ASLCC}^{(k)}) \cdot (\mathbf{U}_{i,ASLCC}^{(k)} - \mathbf{U}_{i,ASLCC}^{(k-1)}) \tag{3.42}$$

Step 11. Update MPTP, $\mathbf{X}_{i,MPTP}^{(k+1)}$, using $\mathbf{U}_{i,ASLCC}^{(k+1)}$ as given in equation (3.43) and update $k = k+1$ and goto Step 3.

$$\mathbf{X}_{i,MPTP}^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k+1)} + \boldsymbol{\sigma}_{\mathbf{X}} \mathbf{U}_{i,ASLCC}^{(k+1)}. \tag{3.43}$$

3.6 Numerical examples

The proposed method is tested on five mathematical and one engineering RBDO examples. Monte Carlo Simulation (MCS) with a sample size of 10^6 is performed at the optimal solution for measuring its reliability for each of the performance functions. The computational efficiency is measured through the number of function calls (nfc) in which f_{FC} is the measure of objective function calls and G_{FC} is the measure of performance function calls during optimization. The results of the proposed method are compared with AMV (Tu et al., 1999), CGA (Ezzati et al., 2015), CC (Yang and Yi, 2009), HCC (Meng et al., 2015), SORA (Du and Chen, 2004), ASORA (Yi et al., 2016), and SLA (Liang et al., 2008) from the literature. In this section, two variants of ASLCC method are presented. In equations (3.36) and (3.37), conjugate gradient and steepest descent search directions are used to calculate MPTP ($\mathbf{X}_{i,MPTP}^{(k+1)}$). The step 4 discussed in Section 3.5.2.2 suggests that when $k \geq 2$, the search direction ($\mathbf{D}_i^{(k)}$) is calculated using equation (3.36), otherwise use equation (3.37). This variant is referred to as ASLCC-2. However, when $\mathbf{D}_i^{(k)}$ is estimated only using equation (3.37) in every iteration, this variant is referred to as ASLCC-1. All the methods are initialized with the same design point and are terminated with the same termination conditions. MATLAB R2016b platform is used for developing the methods and *fmincon* is used as an optimizer that uses sequential quadratic programming method. The termination criterion for *fmincon*

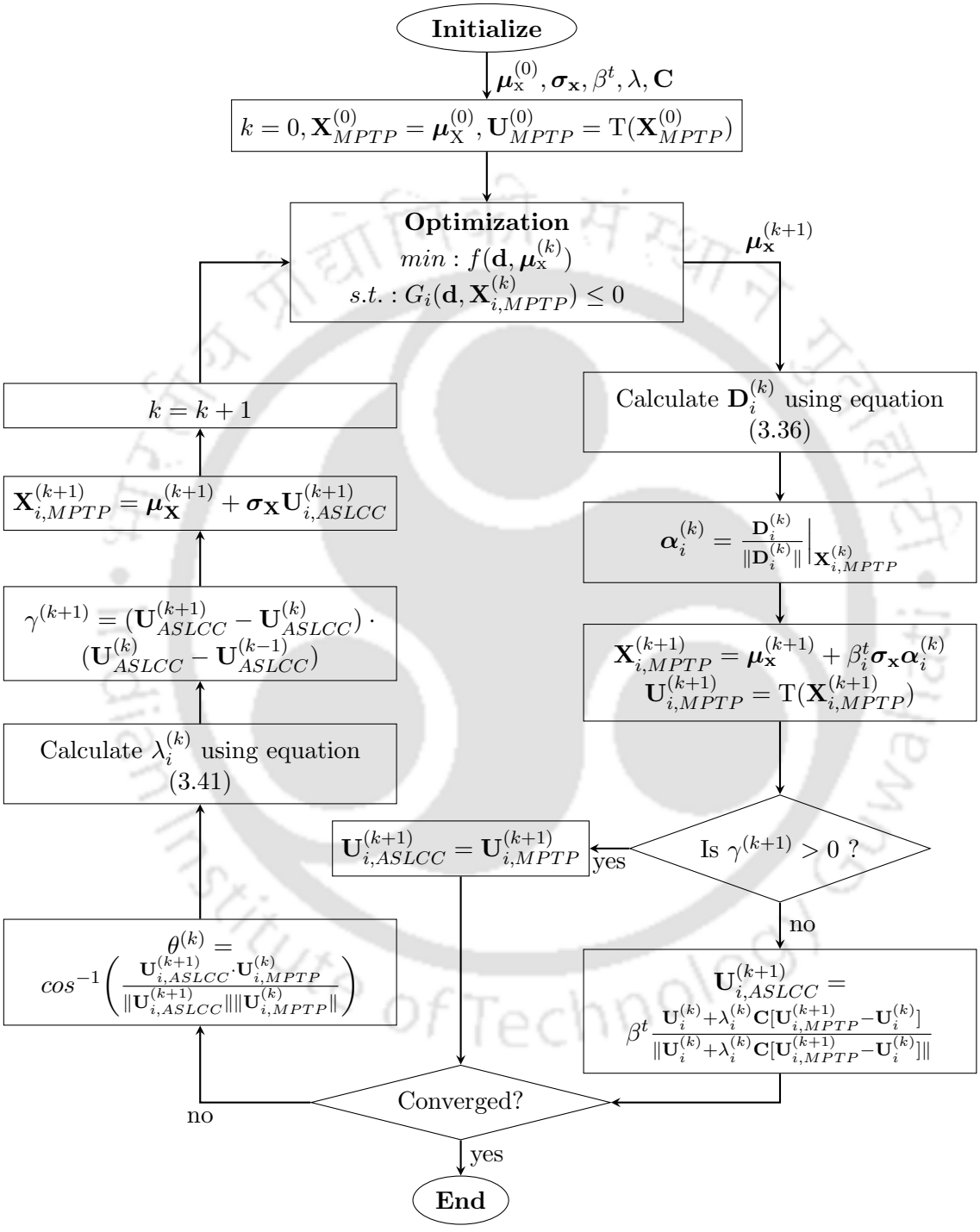


Figure 3.20: Flowchart of ASLCC-2

Table 3.10: RBDO results for example 1 with $\beta^t = 3.5$ ($\lambda = 0.5$)

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc		β_{MCS}^t
			$\mathrm{f_{FC}}$	$\mathrm{G_{FC}}$	(G_1, G_2, G_3)
AMV	-	-	-	-	-
CGA	-1.6409	$(4.5273, 2.1587)$	51	16490	$3.4808, 3.6522$, Inf
CC	-	-	-	-	-
HCC	-	-	-	-	-
SORA	-	-	-	-	-
SLA	-	-	-	-	-
ASORA	-	-	-	-	-
SLSVCG	-1.6595	$(4.5965, 2.1131)$	92	290	$3.4721, 3.5172$, Inf
ASLCC-1	-1.6452	$(4.5402, 2.1483)$	1566	5103	$3.4601, 3.5272$, Inf
ASLCC-2 (Proposed)	-1.6429	$(4.5231, 2.1543)$	218	678	$3.5401, 3.6949$, Inf

is set to the default values, whereas the termination criterion for developed methods is set to $\|f(\mathbf{d}^{(k+1)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k+1)}) - f(\mathbf{d}^{(k)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| / \|f(\mathbf{d}^{(k)}, \boldsymbol{\mu}_{\mathbf{x}}^{(k)})\| \leq 0.001$ or $\|\boldsymbol{\mu}_{\mathbf{x}}^{(k+1)} - \boldsymbol{\mu}_{\mathbf{x}}^{(k)}\| \leq 0.001$.

3.6.1 Example 1

The example presented in section 3.3.1 is considered to demonstrate the effectiveness of the proposed method. The formulation is given in equation (3.15). The initial design point is taken as $\boldsymbol{\mu}_x^{(0)} = [5.0, 5.0]^{\mathrm{T}}$, and the target reliability index for each performance function is considered as $\beta_i^t = 3.5$.

Table 3.10 presents the obtained solutions from the RBDO methods. The objective value (f^*) is corresponding to the obtained optimum solution ($\boldsymbol{\mu}_x^*$). Here, 'β_{MCS}^T' represents the target reliability achieved for each optimum solution as listed in the sixth column of the table. The performance function calls ($\mathrm{G_{FC}}$) denote the total number of times the performance function is called by *fmincon* and for gradient calculation. The term '$\mathrm{f_{FC}}$' represents the total number of objective function calls in the entire optimization process.

It can be seen from the six column of Table 3.10 that ASLCC-2 method generates the reliable solution. However, SLSVCG, CGA and ASLCC-1 methods could not generate any reliable solution. Rest of the methods fail to converge for the given RBDO example. Figure 3.21 shows the optimal solution obtained by ASLCC-2. It can be seen that the performance functions $G_1(\mathbf{X})$ and $G_2(\mathbf{X})$ are active at the solution and the performance function $G_3(\mathbf{X})$ is inactive. The same observation can be seen at the last column of Table 3.10 for $G_3(\mathbf{X})$ at which β_{MCS}^t value becomes infinity.

The computational efficiency of the methods is quantified using the function evaluations that is presented in the fourth and fifth columns of Table 3.10. ASLCC-2 is found better than CGA and ALSCC-1. However, it requires more function evaluation than SLSVCG. ALSCC-1 is also found computationally efficient than CGA method.

The convergence of ALSCC-1, ALSCC-2 and CGA methods is shown in Figure 3.22. A smooth convergence of ASLCC-2 can be seen, whereas ASLCC-1 faces difficulty while converging to the solution. ALSCC-1, thus, requires almost seven time more iterations than ALSCC-2. The convergence of all methods for the performance function ($G_2(\mathbf{X})$) with respect to the number of iterations are shown from Figures 3.23 to 3.31. It can be seen that ALSCC-2 shows smooth convergence while dealing with the concave performance function. Other methods show oscillation because MPTPs are estimated using the steepest descent search directions or by conjugate gradient search direction.

The chaos control factor (λ) plays a vital role in generating the solution for chaos control-based RBDO methods. Therefore, two λ values are considered; one nearer to zero ($\lambda = 0.2$) and the other closer to one ($\lambda = 0.8$). Table 3.11 presents the obtained solutions from these methods. Here, CC and HCC methods achieve convergence for $\lambda = 0.2$ but shows chaos for $\lambda = 0.8$. ASLCC-1 also demonstrates similar chaos for $\lambda = 0.8$. This is because when λ is closer to one, the method estimates MPTPs using the steepest descent search directions with chaos control theory and, therefore, becomes unable. However, ASLCC-2 variant shows no sensitivity towards the initial value of λ. This is because $\mathbf{U}_{i,MPTP}^{(k+1)}$ is estimated using the conjugate gradient search directions. Furthermore, the adaptive nature of λ that depends on the angle condition as presented in equation (3.41) also helps ASLCC-2 to become insensitivity for λ values.

3.6.2 Example 2

This nonlinear RBDO example is the same as presented in section 3.3.2. It consists of a highly nonlinear concave performance function with a nonlinear objective function. The RBDO problem formulation is given in equation (3.16). This example consists of two independent random normal variables with standard deviation of 0.6, and the initial design point is considered as $\boldsymbol{\mu}_{\mathbf{x}}^{(0)} = [5.0, 5.0]^{\mathrm{T}}$.

The RBDO results obtained by different methods are summarized in Table 3.12. It can be seen from the last column of the table that ALSCC-2, SLShV-CG, CGA, CC, and HCC methods generate the reliable solutions. However, SORA is unable to generate any reliable

Table 3.11: RBDO results for example 1 with different values of λ

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc	
			f_{FC}	G_{FC}
$\lambda = 0.2$				
CC	-1.6411	$(4.5281, 2.1582)$	30	37206
HCC	-1.6409	$(4.5274, 2.1589)$	37	12582
ASLCC-1	-	-	-	-
ASLCC-2 (Proposed)	-1.6429	$(4.5231, 2.1543)$	218	678
$\lambda = 0.8$				
CC	-	-	-	-
HCC	-	-	-	-
ASLCC-1	-	-	-	-
ASLCC-2 (Proposed)	-1.6429	$(4.5231, 2.1543)$	218	678

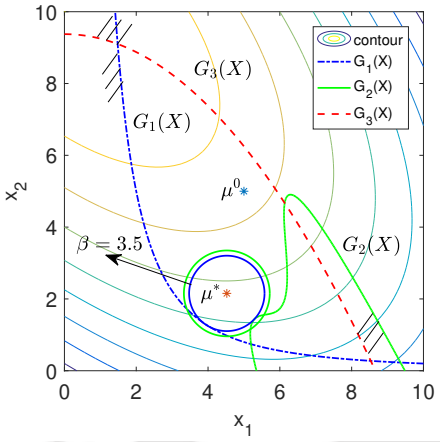


Figure 3.21: Contour plot of example 1 for ASLCC-2

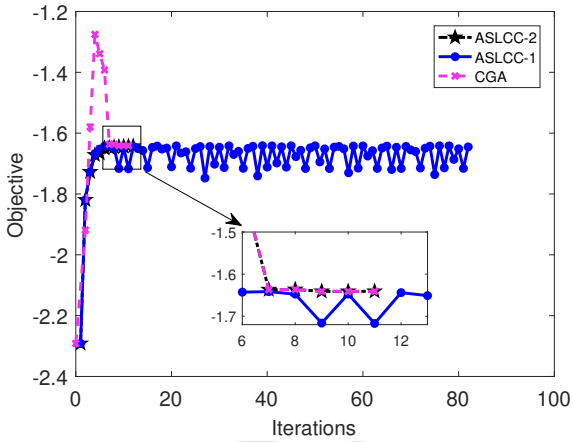


Figure 3.22: The convergence plot for example 1

solution. Among them, ASLCC-2 generates the best reliable solution. The obtained solution by ASLCC-2 method is shown in Figure 3.32. It can be seen that the performance function is active at the solution. The convergence plot is shown in Figure 3.33 in which all methods converge smoothly.

Table 3.12 also presents the computational efficiency in the fourth and fifth columns. ALSCC-2 is found to be the most efficient method followed by HCC and CGA methods.

Table 3.13 presents results for different values of λ for chaos control-based RBDO methods. Among them, ASLCC-2 generates the best reliable solution with less number of function evaluations. Moreover, it is found to be insensitive for different values of λ.

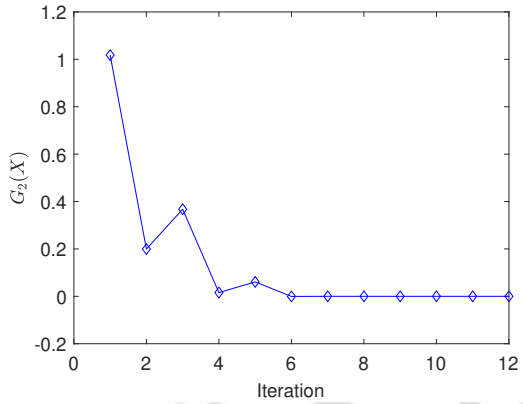


Figure 3.23: ASLCC-2

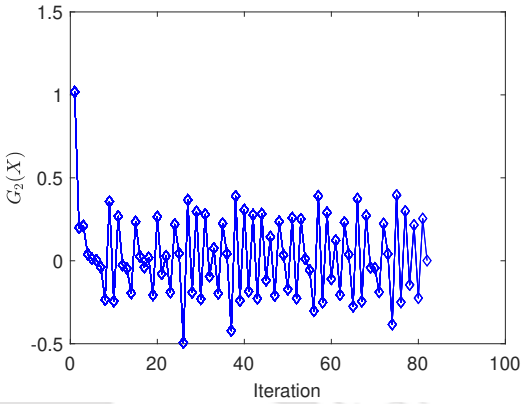


Figure 3.24: ASLCC-1

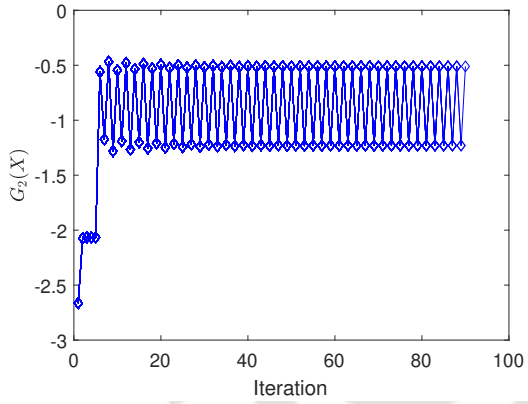


Figure 3.25: AMV

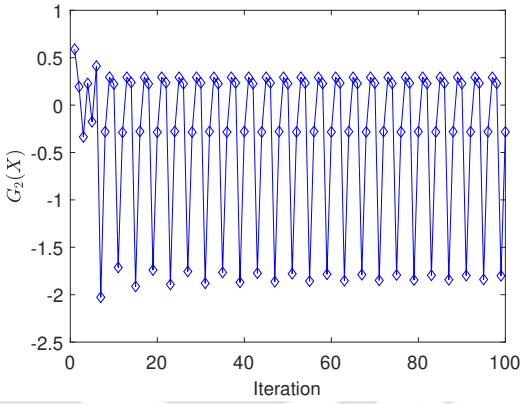


Figure 3.26: ASORA

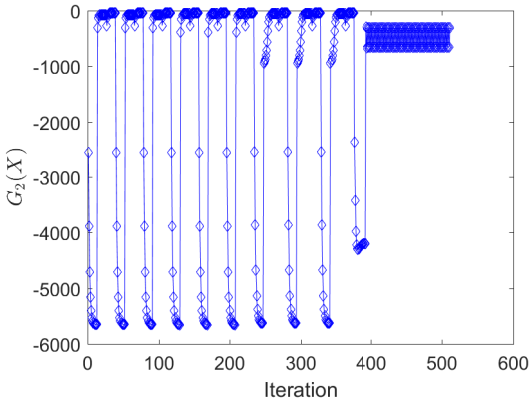


Figure 3.27: CC

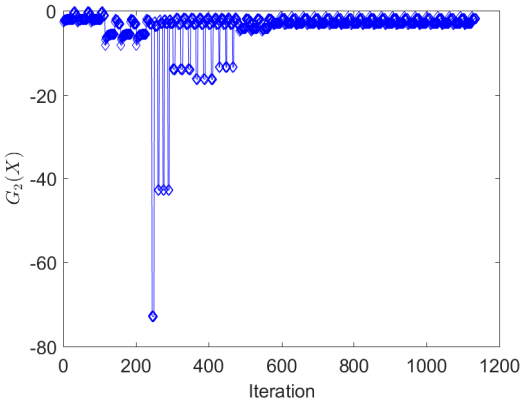


Figure 3.28: CGA

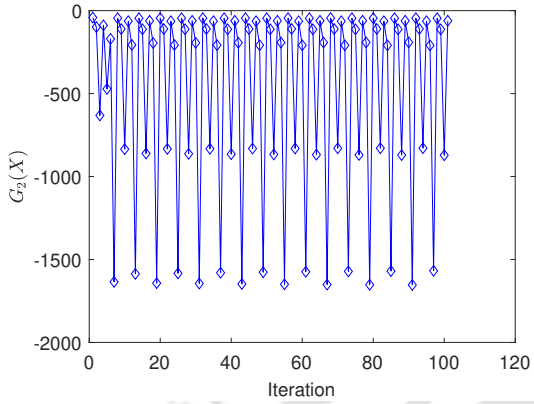


Figure 3.29: SLA

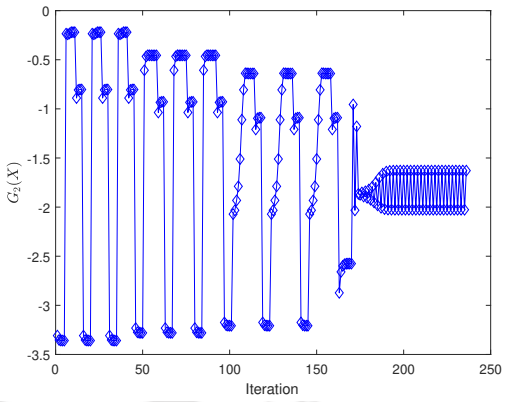


Figure 3.30: HCC

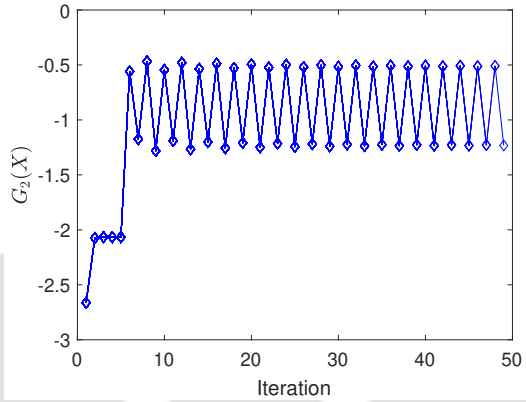


Figure 3.31: SORA

Table 3.12: RBDO results for example 2 with $\beta^t = 3.0$ ($\lambda = 0.5$)

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{x}}$	nfc		β^t_{MCS}
			f_{FC}	G_{FC}	(G_1)
AMV	-	-	-	-	-
CGA	37.3957	$(3.5760, 3.7641)$	33	993	3.0701
CC	37.3932	$(3.5757, 3.7637)$	39	1519	3.0814
HCC	37.3957	$(3.5761, 3.7640)$	42	801	3.0786
SORA	36.7965	$(2.9152, 3.9933)$	190	2176	2.8109
SLA	-	-	-	-	-
ASORA	-	-	-	-	-
ASLCC-1	-	-	-	-	-
ASLCC-2 (Proposed)	37.3927	$(3.6210, 3.7245)$	281	333	3.0843

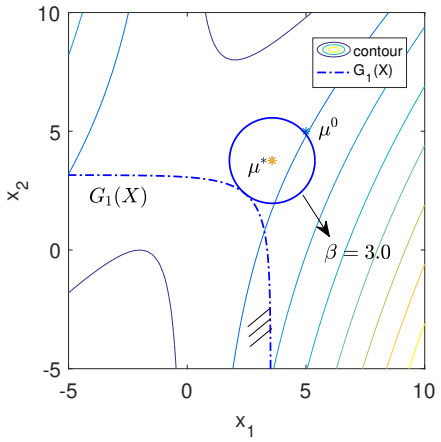


Figure 3.32: Contour plot of example 2 for ASLCC-2

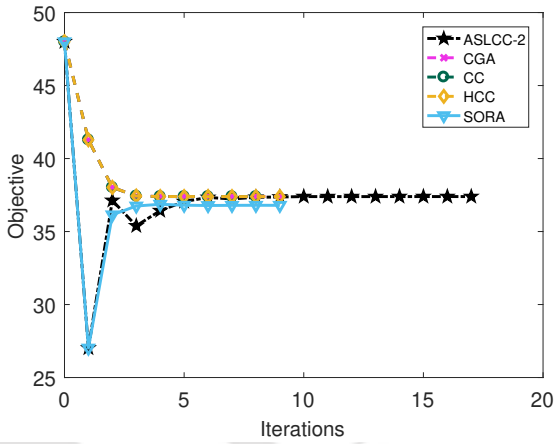


Figure 3.33: The convergence plot for example 2

Table 3.13: RBDO results for example 2 with different values of λ

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc	
			f_{FC}	G_{FC}
$\lambda = 0.2$				
CC	37.3830	(3.5744, 3.7625)	39	3751
HCC	37.3957	(3.5761, 3.7640)	42	1857
ASLCC-1	-	-	-	-
ASLCC-2 (Proposed)	37.3957	(3.5742, 3.7655)	361	411
$\lambda = 0.8$				
CC	37.3957	(3.5761, 3.7640)	39	1235
HCC	37.3957	(3.5761, 3.7640)	42	1519
ASLCC-1	-	-	-	-
ASLCC-2 (Proposed)	37.3955	(3.6140, 3.7309)	302	340

3.6.3 Example 3

The present RBDO example (Jiang et al., 2017b) given in equation (3.44) consists of three highly nonlinear performance functions, two independent random variables with normal distribution, $x_l \sim N(\boldsymbol{\mu}_{x_l}, 0.2^2)$ for $l = 1, 2$. The initial design point is considered as $\boldsymbol{\mu}_x^{(0)} = [1.0, 1.0]^{\mathrm{T}}$ with target reliability of 2.0 for each performance function.

Table 3.14: RBDO results for example 3 with $\beta^t = 2.0$ ($\lambda = 0.5$)

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc		β_{MCS}^t
			$\mathrm{f_{FC}}$	$\mathrm{G_{FC}}$	(G_1, G_2, G_3)
AMV	2.8619	$(0.5, 0.3619)$	27	17316	$2.0456, 2.2938, 3.4503$
CGA	2.8619	$(0.5, 0.3619)$	28	10104	$2.0472, 2.2866, 3.4702$
CC	2.8618	$(0.5, 0.3618)$	26	10257	$2.0486, 2.2886, 3.6153$
HCC	2.8619	$(0.5, 0.3619)$	26	5145	$2.0472, 2.2866, 3.4702$
SORA	2.8619	$(0.5, 0.3619)$	55	5256	$2.0456, 2.2938, 3.4503$
SLA	2.8619	$(0.5, 0.3619)$	80	263	$2.0472, 2.2866, 3.4702$
ASORA	2.8619	$(0.5, 0.3619)$	53	168	$2.0456, 2.2938, 3.4503$
SLSVCG	2.8619	$(0.5, 0.3619)$	54	193	$2.0456, 2.2938, 3.4503$
ASLCC-1	2.8619	$(0.5, 0.3619)$	59	180	$2.0456, 2.2938, 3.4503$
ASLCC-2 (Proposed)	2.8619	$(0.5, 0.3619)$	54	180	$2.0456, 2.2938, 3.4503$

$$
\begin{aligned}
&\text{Find: } [\boldsymbol{\mu}_{x_1}, \boldsymbol{\mu}_{x_2}]^{\mathrm{T}} \\
&\text{min: } \boldsymbol{\mu}_{x_1} + \boldsymbol{\mu}_{x_2}, \\
&\text{s.t.: } P\left[G_i(\mathbf{X}) > 0\right] \leq \phi(-\beta_i^t), \quad i = 1, 2, 3, \\
&\qquad G_1(\mathbf{X}) = 9 - 5\sin(x_1 - 1)^2 - 5x_2 + x_1 x_2, \\
&\qquad G_2(\mathbf{X}) = 11 - 5\cos(-x_1 - 1)^2 - 5x_2 - 1.5x_1 x_2, \\
&\qquad G_3(\mathbf{X}) = -38 - 6\sin(x_1 - 1)^2 + 12x_2 + x_1 x_2, \\
&\qquad 0.5 \leq \boldsymbol{\mu}_{x_l} \leq 4,\ x_l \sim N(\boldsymbol{\mu}_{x_l}, 0.2^2) \text{ for } l = 1, 2, \\
&\qquad \beta_i^T = 2.0,\ \boldsymbol{\mu}_x^{(0)} = [1.0, 1.0]^{\mathrm{T}}.
\end{aligned}
\tag{3.44}
$$

The results of the chosen set of methods are summarized in Table 3.14. It can be seen that all the methods converge to the similar optimal solution and thus, have the same objective function value. The obtained solution is also reliable that can be seen from the columns of 'β_{MCS}^t'. Figure 3.34 shows that the optimum solution obtained by ASLCC-2 satisfies the target reliability for all performance functions. The convergence plot for all methods is shown in Figure 3.35 in which all methods converge smoothly.

Table 3.14 presents the computational efficiency of all the methods in the fourth and fifth columns. It can be seen that ASORA emerges at the most efficient method followed by ASLCC-1 and ASLCC-2. Other methods require many function evaluations.

The results obtained for different initial values of λ are tabulated in Table 3.15. It can be seen that ASLCC-2 is not sensitivity for λ. However, the reliable solution generated by

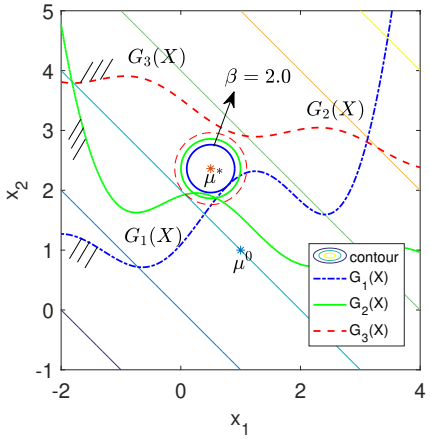


Figure 3.34: Contour plot of example 3 for ASLCC-2

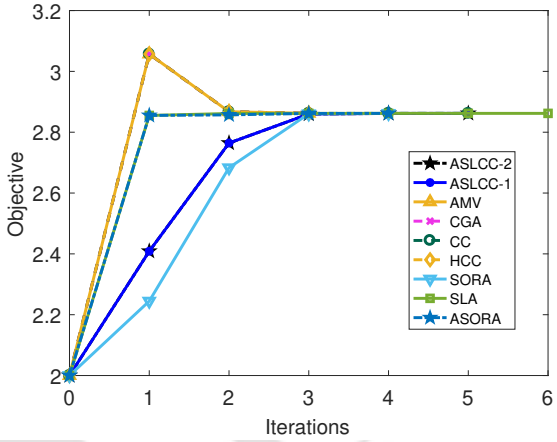


Figure 3.35: The convergence plot for example 3

Table 3.15: RBDO results for example 3 with different values of λ

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc	
			f_{FC}	G_{FC}
$\lambda = 0.2$				
CC	2.8612	$(0.5, 2.3612)$	25	23430
HCC	2.8619	$(0.5, 2.3619)$	27	5832
ASLCC-1	2.8612	$(0.5, 2.3612)$	59	180
ASLCC-2 (Proposed)	2.8618	$(0.5, 2.3618)$	54	180
$\lambda = 0.8$				
CC	2.8619	$(0.5, 2.3619)$	29	5760
HCC	2.8619	$(0.5, 2.3619)$	27	4716
ASLCC-1	2.8619	$(0.5, 2.3619)$	59	180
ASLCC-2 (Proposed)	2.8619	$(0.5, 2.3619)$	54	180

ASLCC-1 and CC methods for $\lambda = 0.2$ has the best objective function value.

Table 3.16: RBDO results for example 4 with $\beta^t = 2.0$ ($\lambda = 0.5$)

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc		β_{MCS}^t
			$\mathrm{f_{FC}}$	$\mathrm{G_{FC}}$	(G_1, G_2)
AMV	1.3038	(2.8163, 3.2769)	48	3054	1.8558,Inf
CGA	1.3034	(2.8149, 3.2789)	53	30470	1.8525,Inf
CC	1.3038	(2.8163, 3.2769)	48	5102	1.8558,Inf
HCC	1.3038	(2.8163, 3.2769)	48	3190	1.8558,Inf
SORA	1.3037	(2.8180, 3.2748)	93	507	1.8586,Inf
SLA	1.3038	(2.8174, 3.2756)	60	225	1.8567,Inf
ASORA	1.3038	(2.8163, 3.2769)	138	283	1.8539,Inf
SLSVCG	1.3038	(2.8163, 3.2769)	56	104	1.8539,Inf
ASLCC-1	1.3038	(2.8162, 3.2770)	175	369	1.8587,Inf
ASLCC-2 (Proposed)	1.3038	(2.8151, 3.2784)	267	583	1.8619,Inf

3.6.4 Example 4

$$
\begin{aligned}
\text{Find: } & [\boldsymbol{\mu}_{x_1}, \boldsymbol{\mu}_{x_2}]^{\mathrm{T}} \\
\text{min: } & (\boldsymbol{\mu}_{x_1} - 3.7)^2 + (\boldsymbol{\mu}_{x_2} - 4)^2, \\
\text{s.t.: } & P\left[G_i(\mathbf{X}) < 0\right] \leq \phi(-\beta_i^t), \quad i = 1, 2, \\
& G_1(\mathbf{X}) = -x_1 \sin(4x_1) - 1.1 x_2 \sin(2x_2), \\
& G_2(\mathbf{X}) = x_1 + x_2 - 3, \\
& 0 \leq \boldsymbol{\mu}_{x_l} \leq 3.7, \; x_l \sim N(\boldsymbol{\mu}_{x_l}, 0.1^2) \text{ for } l = 1, 2, \\
& \beta_i^t = 2.0, \; \boldsymbol{\mu}_x^{(0)} = [1.0, 1.0]^{\mathrm{T}}.
\end{aligned}
\tag{3.45}
$$

The RBDO model (Chen et al., 2013b) given in equation (3.45) is another highly nonlinear example that consists of two random normal variables and two performance functions. The target reliability for each performance function is considered as 2.0 and the initial design point is considered as $\boldsymbol{\mu}_x^{(0)} = [1.0, 1.0]^{\mathrm{T}}$.

Table 3.16 presents the results obtained from the chosen set of methods. ASLCC-2 shows the closest value of β_{MCS}^t with respect to the desired target reliability of 2.0. The obtained solution by ALSCC-2 is shown in Figure 3.36. It can be seen that the current solution is unable to satisfy the performance function $G_1(\mathbf{X})$ due to its non-linearity. The performance function $G_2(\mathbf{X})$ is inactive and therefore, the last column of Table 3.16 shows infinity value. Figure 3.37 shows smooth progress of all the methods for the given example.

Table 3.16 presents the computational complexity in the fourth and fifth columns. SLA is found to be the most efficient method for this example followed by ASORA, ASLCC-1 and

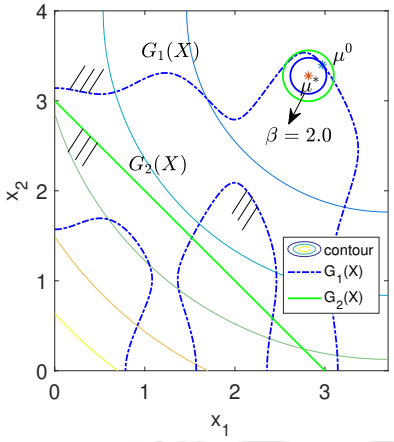


Figure 3.36: Contour plot of example 4 for ASLCC-2

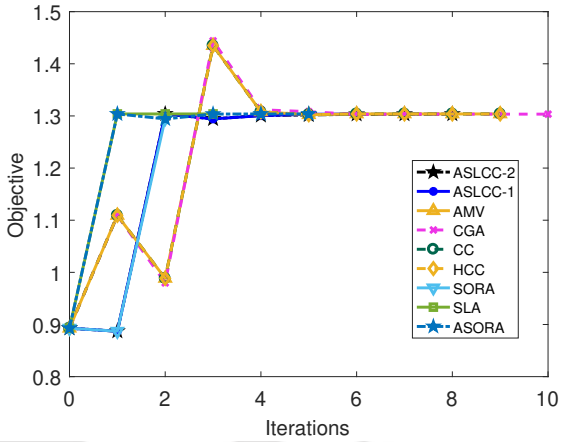


Figure 3.37: The convergence plot for example 4

Table 3.17: RBDO results for example 4 with different values of λ

Methods	f^*	$\boldsymbol{\mu}_x^*$	nfc	
			f_{FC}	G_{FC}
$\lambda = 0.2$				
CC	1.3038	(2.8163, 3.2770)	48	21158
HCC	1.3038	(2.8163, 3.2769)	48	3182
ASLCC-1	1.3038	(2.8177, 3.2753)	133	281
ASLCC-2 (Proposed)	1.3038	(2.8153, 3.2781)	267	583
$\lambda = 0.8$				
CC	1.3038	(2.8163, 3.2769)	48	5102
HCC	1.3038	(2.8163, 3.2769)	48	3022
ASLCC-1	1.3038	(2.8163, 3.2769)	164	343
ASLCC-2 (Proposed)	1.3038	(2.8153, 3.2781)	267	583

ASLCC-2. However, the solution generated by ALSCC-2 is found to be more closer to the desired target reliability for the performance functions.

Table 3.17 presents the RBDO results for different values of λ for choas control-based methods. ASLCC-2 requires the same number of function evaluations, whereas the computational efficiency of ASLCC-1, HCC, and CC methods depend on λ values.

3.6.5 Example 5

The RBDO model of this example is the same as presented in equation (3.18) that has 10 statistically independent random variables with normal distribution. There are eight performance

Table 3.18: RBDO results for example 5 with $\beta^t = 3.0$ ($\lambda = 0.5$)

Methods	f^*	$\boldsymbol{\mu}_{\mathbf{x}}^*$	nfc	
			f_{FC}	G_{FC}
AMV	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	184	666272
CGA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	205	541500
CC	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	197	2827712
HCC	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	197	666272
SORA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	496	17031
SLA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	382	2337
ASORA	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	376	2159
ASLCC-1	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	492	3343
ASLCC-2 (Proposed)	27.7466	(2.1350, 2.3309, 8.7094, 5.1021, 0.9225, 1.4452, 1.3885, 9.8094, 8.1556, 8.4755)	489	3343

	β_{G_1}	β_{G_2}	β_{G_3}	β_{G_4}	β_{G_5}	β_{G_6}	β_{G_7}	β_{G_8}
All methods	3.0280	3.0045	2.9781	2.9824	3.0000	Inf	3.0332	Inf

functions with target reliability index of 3.0.

Table 3.18 presents the results of all methods. It can be seen from the columns of β_{MCS}^t that none of the methods generate any reliable solution for this example. Moreover, these methods converge to the same solution as that of ASLCC-2 with minimum target reliability of 2.9781 for $G_3(\mathbf{X})$. Figure 3.38 shows convergence of all the methods. It can be seen that SLA, ASORA, ASLCC-1, ASLCC-2, and SORA require less number of iterations to converge to their respective solutions as compared to the other methods.

The computational efficiency of the methods is presented in the fourth and fifth columns of Table 3.18. It is found that ASORA needs the least number of function evaluations followed by SLA, ASLCC-2, and ASLCC-1. Rest of the methods need many function evaluations.

The results obtained for different values of λ are presented in Table 3.19. It can be seen that all chaos control-based methods converge to the same solution. ASLCC-2 needs the same number of function evaluations for both values of λ. However, CC and HCC methods require relatively large number of function evaluations.

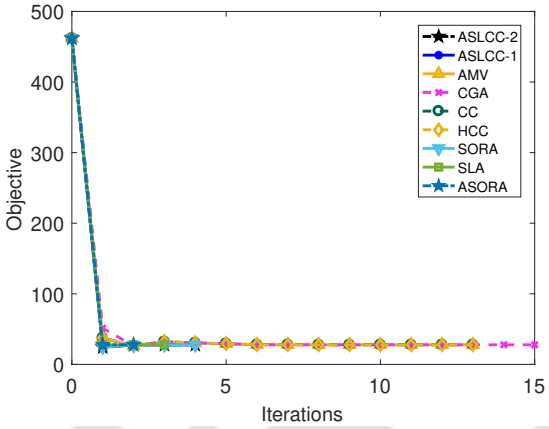


Figure 3.38: Convergence plot of example 5

Table 3.19: RBDO results for example 5 with different values of λ

Methods	f^*	nfc	
		f_{FC}	G_{FC}
$\lambda = 0.2$			
CC	27.7466	197	7085792
HCC	27.7466	197	666272
ASLCC-1	27.7466	509	3471
ASLCC-2 (Proposed)	27.7466	489	3343
$\lambda = 0.8$			
CC	27.7466	197	1438752
HCC	27.7466	197	666272
ASLCC-1	27.7466	528	3623
ASLCC-2 (Proposed)	27.7466	489	3343

3.6.6 Speed reducer example

A speed reducer is used to rotate the propeller of a light plane. It is the same example that is shown by equation (3.19). The objective is to reduce the weight of the speed reducer, which is subjected to 11 performance functions. The RBDO example consists of seven independent random variables. The RBDO problem formulation is given in equation (3.19).

Table 3.20 presents the results obtained by all the methods. It can be seen that all methods generate equivalent optimal solution as that of ASLCC-2 with similar reliability. All the methods converge to the same solution with same target reliability. The performance functions $G_5(\mathbf{X}), G_6(\mathbf{X}), G_8(\mathbf{X})$ and $G_{11}(\mathbf{X})$ are only the active constraints at the optimum solution. The progress of solutions is shown Figure 3.39, which indicates smooth convergence for all the

Table 3.20: RBDO results for speed reducer example with $\beta^t = 3.0(\lambda = 0.5)$

Methods	f^*	$\boldsymbol{\mu}_{\mathbf{x}}^*$	nfc	
			f_{FC}	G_{FC}
AMV	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	229680
CGA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	120240
CC	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	976272
HCC	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	48	229680
SORA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	14874
SLA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	1038
ASORA	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	112	1520
SLSVCG	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	77	1038
ASLCC-1	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	79	1014
ASLCC-2 (Proposed)	3038.612	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	76	1014

	β_{G_1}	β_{G_2}	β_{G_3}	β_{G_4}	β_{G_5}	β_{G_6}	β_{G_7}	β_{G_8}	β_{G_9}	$\beta_{G_{10}}$	$\beta_{G_{11}}$
All methods	Inf	Inf	Inf	Inf	3.0057	3.0052	Inf	3.0002	Inf	Inf	2.9953

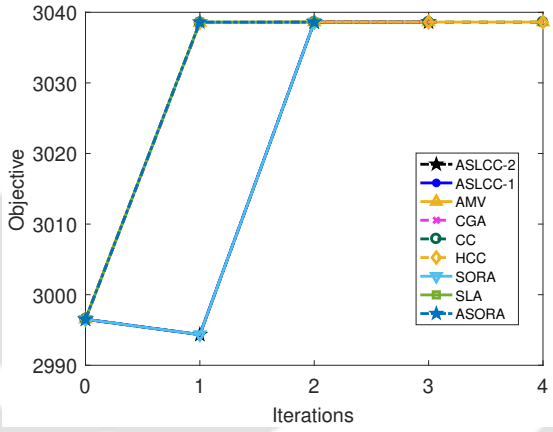


Figure 3.39: Convergence plot of speed reducer example

methods.

The computational efficiency is also presented in Table 3.20. It can be seen from the fourth and fifth columns of the table that ASLCC-1 and ASLCC-2 are the most efficient methods followed by SLA and ASORA. Rest of the methods require relatively many function evaluations.

The results of different initial values of λ for chaos control-based methods are listed in Table 3.21. All methods converge to the same solution. The computational efficiency of ASLCC-1 and ASLCC-2 methods does not depend on the initial chaos control factor for the given example. However, the efficiency of CC method changes drastically as the value of λ increases. HCC method seems to be insensitive toward the initial value of λ for the given example.

Table 3.21: RBDO results for speed reducer example with different values of λ

Methods	f^*	nfc	
		f_{FC}	G_{FC}
$\lambda = 0.2$			
CC	3038.612	48	2439888
HCC	3038.612	48	229680
ASLCC-1	3038.612	79	1014
ASLCC-2 (Proposed)	3038.612	76	1014
$\lambda = 0.8$			
CC	3038.612	48	488400
HCC	3038.612	48	229680
ASLCC-1	3038.612	79	1014
ASLCC-2 (Proposed)	3038.612	76	1014

3.7 Closure

An approximate single-loop chaos control method (ASLCC-2) using conjugate gradient search directions was proposed. The aim was to track oscillation among MPTPs while solving various RBDO examples. In the proposed method, MPTPs were estimated using conjugate gradient search directions in every iteration. When these points started oscillating, chaos control theory was used to update the current MPTP. The oscillation criterion has also been proposed that estimated chaos among three consecutive MPTPs in the standard normal space. Among the chosen set of RBDO methods, ASLCC-2 generated the best reliable solutions for all examples. The convergence plots suggested a smooth convergence of ASLCC-2 for all examples, especially for Example 3.6.1 and Example 3.6.2 when oscillation among MPTPs was observed with rest of the methods. ALSCC-2 was also computationally efficient that was estimated using the number of function evaluations required to generate the reliable solution. It was found computationally efficient in four out of six examples. The proposed method was also compared with other chaos control-based methods for different initial values of λ. ASLCC-2 was found to be insensitive for different λ values while solving all examples.

Although, ASLCC-2 demonstrated good convergence for example where oscillations ware observed, it could converge to a local optimal solution. Implementation of global optimizer is required to make the method more robust. In the next chapter, a meta-heuristic based global optimizer named differential evolution will be used to solve RBDO problems.

Chapter 4

Single-loop RBDO using Differential Evolution

The single-loop methods discussed in chapter 3 use numerical optimization techniques that are prone to converge to a locally optimal solution (Deb et al., 2009, Lobato et al., 2017). These methods have a tendency to oscillate and often diverge from the solution (Meng et al., 2015, Yang and Yi, 2009). One of the remedies to this convergence issue is to use meta-heuristic algorithms, which are found to be effective in solving RBDO problems. Most of these algorithms are population based and are computationally expensive. Moreover, solving RBDO problems using meta-heuristic algorithms with traditional double-loop method adds up the computational cost. Therefore, there is a need to incorporate single-loop method with meta-heuristic algorithms.

The single-loop methods can be improved by incorporating approaches that can help in generating reliable solution. The use of meta-heuristic algorithms can enhance searching power of RBDO methods so that they can become global optimizer for generating reliable and near optimal solutions. Therefore, third objective of the thesis aims to develop a single-loop RBDO formulation using shifting vector approach (refer equation (3.7)) that can give direction to violated constraints/performance functions towards the feasible direction for generating reliable solution. The formulation also includes target and trial vectors of DE for guiding the algorithm. A heuristic parameter is designed to execute two different mutation operators of DE for exploration and exploitation in the search space. In the following sections, differential evolution (DE) is explained followed by the proposed formulation and adaptive DE.

4.1 Differential evolution

Differential evolution (DE) (Storn and Price, 1997) is one of the successful meta-heuristic algorithms for global optimization. DE is a population-based algorithm that works with a set of vectors. It inherits a parallel-search space property for escaping the local minima. DE starts with a random population of vectors in which each vector represents a set of decision variables for the given problem. These vectors are referred to as target vectors, $\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$, in which t represents $t-$th target vector and k represents $k-$th generation counter. Since DE is used for solving RBDO problem, the notation for vector is kept the same as the mean value of random variable. In the standard loop of generations, each target vector ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$) is mutated and referred to as a mutant vector ($\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t)$), which is given as

$$\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t) = \boldsymbol{\mu}_{\mathbf{r}_1}^{(k)}(t) + F \times (\boldsymbol{\mu}_{\mathbf{r}_2}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r}_3}^{(k)}(t)), \tag{4.1}$$

where $\mathbf{r_1} \neq \mathbf{r_2} \neq \mathbf{r_3}$ are randomly chosen vectors from the current population, and F is the scaling factor. Thereafter, a trial vector ($\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t)$) is created for each target vector ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$), which is given as

$$\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j) = \begin{cases} \boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j) & \text{if } r \leq p_c \text{ or } j = rnbr(i), \\ \boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t_j) & \text{if } r > p_c \text{ and } j \neq rnbr(i), \end{cases} \tag{4.2}$$

where subscript j with t in $\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t_j)$, $\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j)$, and $\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j)$ represents $j-$th component of target, mutant and trial vectors, respectively. r is a random number between 0 to 1, p_c is the crossover rate, and $rnbr(i)$ is a randomly chosen index $\in \{1, 2, \ldots, n\}$, which ensures that $\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j)$ gets at least one component from $\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j)$. The first DE implementation (Storn and Price, 1997) uses greedy selection after creation of trial vectors in which the target vector for the next generation ($\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}(t)$) is created as

$$\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}(t) = \begin{cases} \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t), & \text{if } F(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t)) < F(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)), \\ \boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t), & \text{otherwise}, \end{cases} \tag{4.3}$$

where $F(\cdot)$ represents fitness of a vector for minimization problem. The loop over generations continues till the termination condition gets satisfied.

4.2 The Proposed Method and it's Implementation

4.2.1 Single-loop formulation using shifting vector approach and DE vectors

The formulation for single-loop method was given in equation (3.6) in which KKT conditions were used to approximate reliability analysis. The modified single-loop formulation using shifting vector approach is given as

$$\begin{aligned}
&\text{Min. } f(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{X}}),\\
&\text{s.t.: } G_i(\mathbf{d}, \boldsymbol{\Psi}^{(k)}) \leq 0, \qquad\qquad i = 1, \ldots, I,\\
&\text{where } \boldsymbol{\Psi}^{(k)} = \begin{cases} \mathbf{X}^{(k)}_{i,MPTP}, & \forall \text{ target vectors},\\ \boldsymbol{\mu}^{(k+1)}_{\mathbf{U}} - \mathbf{S}^{(k+1)}_i, & \forall \text{ trial vectors}, \end{cases}\\
&\mathbf{S}^{(k+1)}_i = \boldsymbol{\mu}^{(k)}_{\mathbf{X}} - \mathbf{X}^{(k)}_{i,MPTP},\\
&\mathbf{X}^{(k)}_{i,MPTP} = \boldsymbol{\mu}^{(k)}_{\mathbf{X}} + \beta^t_i \boldsymbol{\sigma}_{\mathbf{X}} \boldsymbol{\alpha}^{(k)}_{i,\mathbf{X}},\\
&\boldsymbol{\mu}^{(\mathrm{L})}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}^{(\mathrm{U})}_{\mathbf{X}}, \mathbf{d}^{(\mathrm{L})} \leq \mathbf{d} \leq \mathbf{d}^{(\mathrm{U})}
\end{aligned} \tag{4.4}$$

Here, $\mathbf{S}^{(k+1)}_i$ is the shifting vector of $i-$th performance function at $(k+1)-$th iteration. It can be seen that $G_i(\mathbf{d}, \boldsymbol{\Psi}^{(k)})$ includes both $\mathbf{X}^{(k)}_{i,MPTP}$ and $(\boldsymbol{\mu}^{(k+1)}_{\mathbf{U}} - \mathbf{S}^{(k+1)}_i)$. $\mathbf{X}^{(k)}_{i,MPTP}$ is used when evaluating the population of target vectors, whereas $(\boldsymbol{\mu}^{(k+1)}_{\mathbf{U}} - \mathbf{S}^{(k+1)}_i)$ is used for shifting the violated constraint/performance function towards the feasible direction for the population of trial vectors. This shift can help DE in guiding the vectors for better convergence.

4.2.2 Adaptive mutation scheme

Different variants of DE are available in the literature. For example, Section 4.1 presented "DE/rand/1/bin" in which 'rand' represents $\mathbf{r}_1$ in equation (4.1), '1' represents a single difference of vectors $(\boldsymbol{\mu}^{(k)}_{\mathbf{r}_2}(t) - \boldsymbol{\mu}^{(k)}_{\mathbf{r}_3}(t))$ in the equation, and 'bin' represents crossover according to binomial experiments in equation (4.2). This variant of DE has a good capability of exploring the search space for locating the optima. However, the convergence of DE can be slower due to $\mathbf{r}_1$ because the mutation is done with respect to a random vector. The other variant such as "DE/best/1/bin" can be faster since it uses the 'best' vector $(\boldsymbol{\mu}^{(k)}_{\mathbf{best}}(t))$ in terms of fitness in place of a random vector $(\boldsymbol{\mu}^{(k)}_{\mathbf{r}_1}(t))$ in equation (4.1). However, this variant can converge to a local optima since all target vectors are mutated with respect to only the 'best' vector in the current generation (k). Since both the variants have their own merits and demerits, a

heuristic convergence parameter (ζ) is proposed that helps DE to use either of these variants. The parameter ζ is defined as the difference between the ratio of the best fitness to the worst fitness of the target vectors in the current generation (k) to its previous generation ($k-1$), that is,

$$\zeta = \frac{f_{best}^{(k)}}{f_{worst}^{(k)}} - \frac{f_{best}^{(k-1)}}{f_{worst}^{(k-1)}}. \tag{4.5}$$

The adaptive mutation thus can be written as

$$\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t) = \begin{cases} \boldsymbol{\mu}_{\mathbf{r1}}^{(k)}(t) + F \times (\boldsymbol{\mu}_{\mathbf{r2}}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r3}}^{(k)}(t)), & \zeta > \epsilon, \\ \boldsymbol{\mu}_{\mathbf{best}}^{(k)}(t) + F \times (\boldsymbol{\mu}_{\mathbf{r2}}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r3}}^{(k)}(t)), & \text{otherwise}, \end{cases} \tag{4.6}$$

where ϵ is a user defined parameter. The condition ($\zeta > \epsilon$) signifies that mutation of "DE/rand/1/bin" variant can help DE in exploring the search space in the beginning. After some generations, mutation of 'DE/best/1/bin" variant can exploit the promising search space for faster locating the optima.

4.2.3 Constraint handling with DE

The penalty-parameter less approach (Deb, 2000) is used for handling the constraints in which the fitness of each vector in a population of DE is calculated as

$$F(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)) = \begin{cases} f(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)), & \text{if } \boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t) \text{ is feasible;} \\ f_{max} + \sum_{i=1}^{I} \mathrm{CV}_i, & \text{Otherwise.} \end{cases} \tag{4.7}$$

Here, $f(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t))$ is the objective function value, f_{max} is the objective function value of the worst feasible vector in the current population, and CV_i represents constraint violation of $i-$the performance function.

4.2.4 Selection scheme for next generation target vectors

The $(\mu + \lambda)-$strategy is used for selecting the next generation target vectors, instead of using equation (4.3). In this strategy, the target vectors and trail vectors are combined ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)} \bigcup \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}$). The best N vectors are selected based on the fitness values, and these vectors are copied to $\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}$.

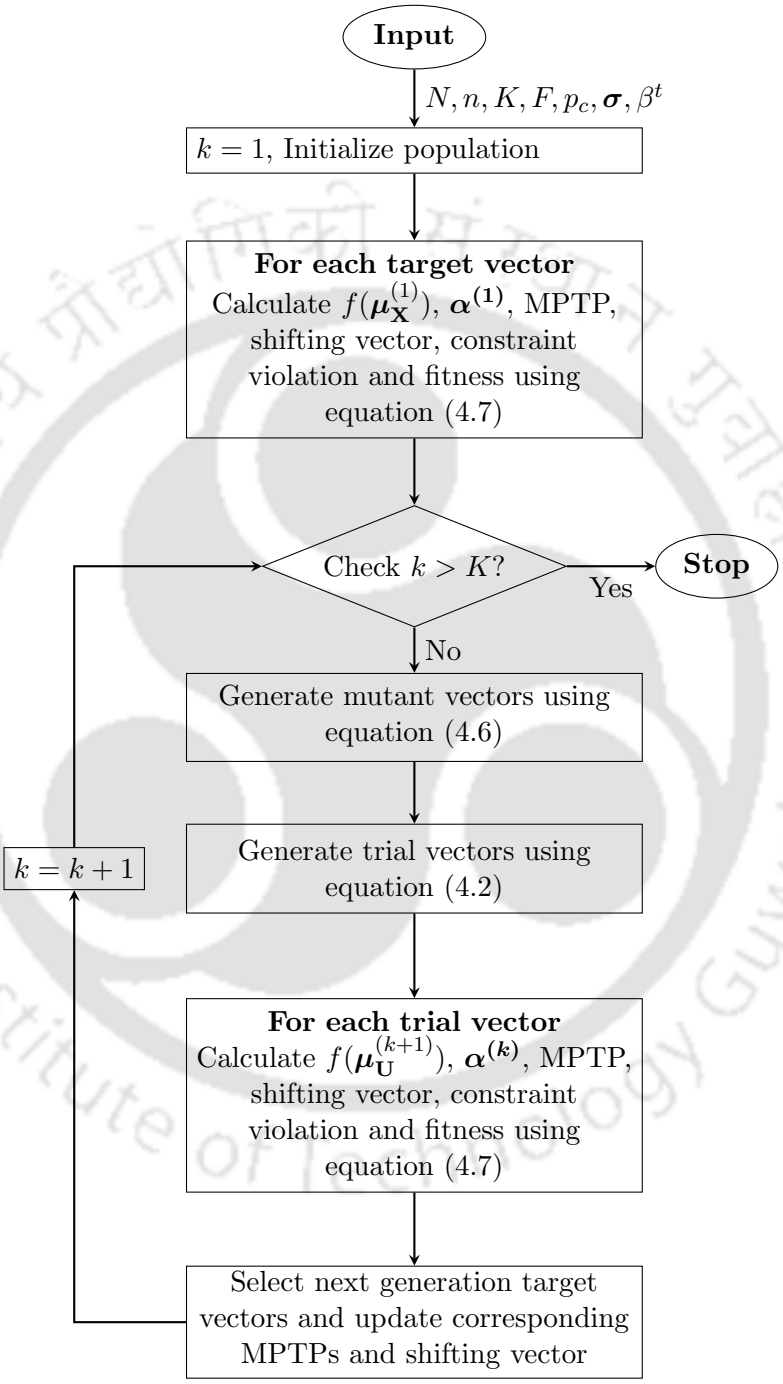


Figure 4.1: Flowchart of SLADE

4.2.5 Steps for implementation

In this section, the procedure for implementing the proposed method is presented. As the proposed method uses a single-loop structure using adaptive differential evolution, it is abbreviated as SLADE. In SLADE, the reliability analysis loop is replaced by approximate MPTP. Figure 4.1 shows the flowchart of SLADE. It can be seen that after initialization, for each target vector, equation (4.7) is used for calculating objective function using the random variables and fitness is assigned using MPTP. The standard loop over number of generations (k) then starts in which mutant and trail vectors are generated. Again, for each trial vector, objective function is calculated and fitness is assigned using equation (4.7). Thereafter, the next generation target vector is constructed, and MPTPs and shifting vectors are updated. The loop continues until the termination criterion is met. The complete details of every step for implementing SLADE are as follows.

Step 1 **Input**: population size (N), number of variables (n), total number of generations (K), scaling factor (F), probability of crossover (p_c), standard deviation ($\boldsymbol{\sigma}$) for random variables, and target reliability index for constraints (β^t), generation counter ($k = 1$).

Step 2 Initialize random population ($P(k)$).

Step 3 For each target vector ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$) of ($P(k)$)

Step 3.1 Calculate objective function, $f(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t))$.

Step 3.2 Calculate the steepest descent direction, $\boldsymbol{\alpha}^{(k)} = \frac{\boldsymbol{\sigma}\nabla G}{\|\boldsymbol{\sigma}\nabla G\|}\Big|_{\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)}$.

Step 3.3 Calculate MPTP for each performance function (i) using equation (4.4) and estimate shifting vector $\mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{X}}}^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \mathbf{X}_{i,MPTP}^{(k)}$.

Step 3.4 Calculate the fitness using equation (4.7). It is noted that constraint violation is calculated for each performance function, $G_i(\mathbf{d}, \mathbf{X}_{i,MPTP}^{(k)})$.

Step 4 If ($k > K$), terminate. Otherwise, continue to Step 5.

Step 5 Generate mutant vectors ($\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}$) using the proposed modified scheme as given in equation (4.6).

Step 6 Generate trial vectors ($\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}$) as given in equation (4.2).

Step 7 For each trail vector

Step 7.1 Calculate objective function, $f(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t))$.

Step 7.2 Calculate the steepest descent direction, $\boldsymbol{\alpha}^{(k)} = \frac{\boldsymbol{\sigma}\nabla G}{\|\boldsymbol{\sigma}\nabla G\|}\Big|_{\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t)}$.

Step 7.3 Calculate MPTP ($\hat{\mathbf{X}}_{i,MPTP}^{(k+1)}$) and shifting vector for each performance function (i) using equation (4.4) and $\mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{U}}}^{(k+2)} = \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \hat{\mathbf{X}}_{i,MPTP}^{(k+1)}$ and estimate constraint violation of $G_i(\mathbf{d}, \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{X}}}^{(k+1)})$.

Step 7.4 Calculate the fitness using equation (4.7).

Step 8 Select target vectors ($\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}$) for the next generation using $(\mu + \lambda)-$strategy. It is noted that the corresponding MPTPs and shifting vector are stored for utilizing them in Step 7.3. Set $k = k + 1$, and go to Step 4.

4.3 Numerical Examples

In this section, three mathematical examples and four engineering examples are solved to demonstrate the accuracy and computational efficiency of SLADE. The optimal solution is evaluated through Monte Carlo Simulation (MCS) with a sample size of 1 million to check its reliability for the performance functions. The results of SLADE are compared with DLM-PMA (Tu et al., 1999), CC (Yang and Yi, 2009), SORA (Du and Chen, 2004), and SLA (Liang et al., 2008) from the literature. The termination condition for these numerical optimization based methods is taken as $\|\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)} - \boldsymbol{\mu}_{\mathbf{X}}^{(k)}\| \leq 0.001$. Further, a double-loop reliability-based design optimization with differential evolution (DLRBDO-DE) is also used for comparison with SLADE. DLRBDO-DE employs PMA for reliability analysis and canonical DE for optimization as discussed in Section 4.1. The parameters of SLADE and DLRBDO-DE are as follows: the scaling factor (F) is taken as 0.3, the crossover probability (p_c) is 0.9, the population size (N) is 50 $\times$ number of variables (n), and the total number of generations (K) is 50 for the first three examples, and 100 for all the engineering examples. The user defined parameter ϵ in equation (4.6) is taken as 10^{-4}. SLADE is run 50 times with a different initial population. Statistical results such as mean, best, worst, and standard deviation are obtained on the objective function value for 50 runs to access the performance.

4.3.1 Example 1

The example discussed in section 3.3.3 is taken as the first RBDO example for evaluating the performance of SLADE. The formulation is given in equation (3.17). The initial design point

Table 4.1: RBDO results for Example 1 with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC	β^T_{MCS}
			G_{FC}	G_1, G_2, G_3
DLM-PMA	6.7219	(3.4363, 3.2855)	5193	2.9987, 3.0307,Inf
CC($\lambda = 0.2$)	6.7254	(3.4391, 3.2863)	12321	2.9839, 3.0673,Inf
SORA	6.7226	(3.4369, 3.2857)	1137	2.9989, 3.0617,Inf
SLA	6.7257	(3.4391, 3.2866)	404	2.9803, 3.1086,Inf
Kriging-SLA[1]	6.7277	(3.4390, 3.2887)	–	2.9760, 3.0814,Inf
SSRBDO[2]	6.7430	(3.4580, 3.2850)	–	3.0357, 3.0091,Inf
CEA[3]	6.7257	(3.4391, 3.2866)	–	2.9889, 3.0258,Inf
DLRBDO-DE	6.7348	(3.4225, 3.3123)	2571706	2.9867, 3.1214,Inf
SLADE (Proposed)	6.7206	(3.4406, 3.2800)	76500	3.0045, 3.0138,Inf

1 Zhang et al. (2020), 2 Li et al. (2019), 3 Chen et al. (2014)

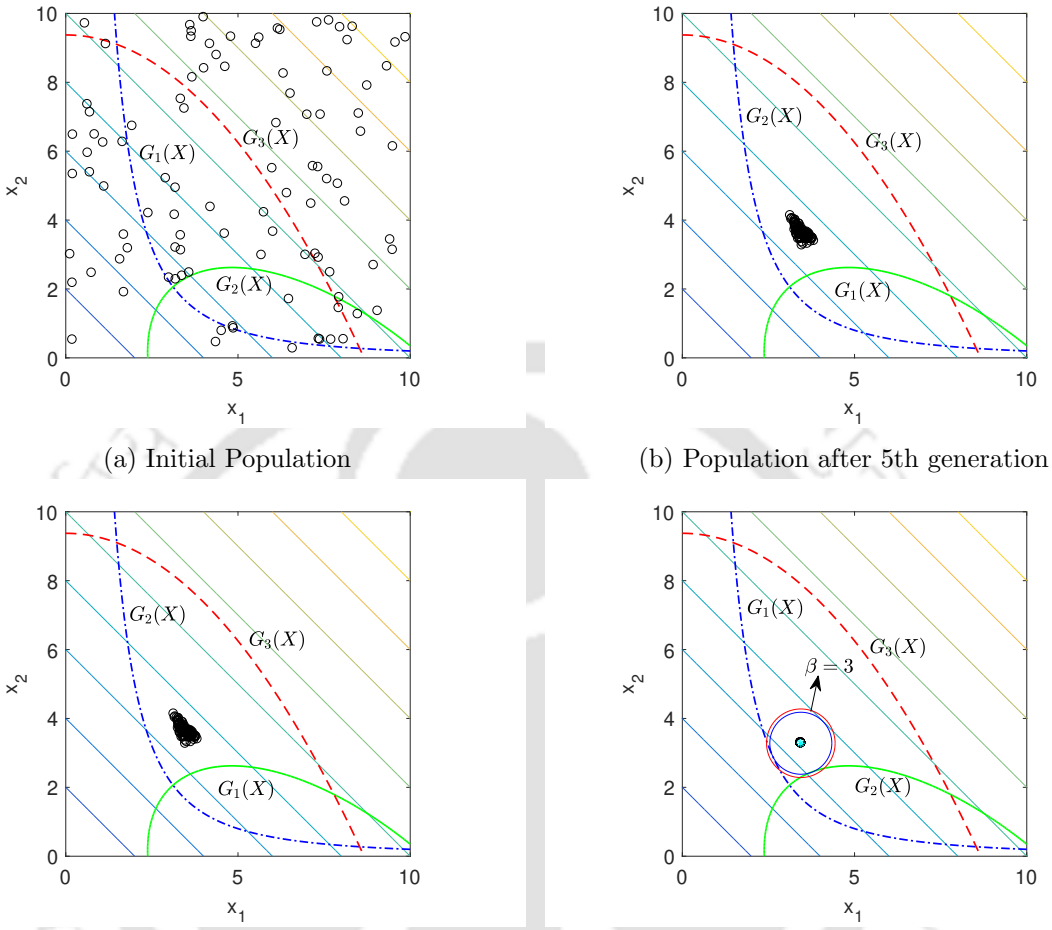
Table 4.2: Statistical values obtained by SLADE for Example 1

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	6.7441	6.7206	6.7220	0.0047

is taken as $\boldsymbol{\mu}_X^{(0)} = [5.0, 5.0]^T$ for DLM-PMA, CC, SORA, and SLA. The target reliability index for all performance functions is considered as $\beta_i^t = 3.0$.

Table 4.1 presents the results obtained by different RBDO methods. It can be seen that the objective function value (f^*) obtained by the proposed SLADE is better than rest of the methods. Also, it generates the reliable solution by achieving the target reliability for performance functions that can be seen at the fifth column of the table. SSRBDO is also able to obtain the reliable solution but with larger objective function value. On the other hand, methods like DLM-PMA, CC, SORA, SLA, Kriging-SLA, CEA and DLRBDO-DE are unable to obtain any reliable solution.

The fourth column of Table 4.1 presents the number of function evaluations (NFC). It can be seen that SLADE requires many function evaluations than DLM-PMA, CC, SORA, and SLA because SLADE performs optimization using population-based DE. When comparing both DE-based RBDO methods, it can be seen that SLADE requires at least 33 times less function evaluations than DLRBDO-DE. Since SLADE is a meta-heuristic algorithm, it is run for 50 times with different initial populations and the statistical values of objective function are presented in Table 4.2. It can be seen that the standard deviation among the obtained objective function values is quite less, thereby suggesting consistent performance of SLADE on the given example.



(a) Initial Population

(b) Population after 5th generation

(c) Population after 10th generation

(d) Population after 25th generation

Figure 4.2: Progress of solutions in different generations for Example 1.

Figure 4.2 shows the progress of solutions/vectors in the variable space at different generations. It can be observed that the solutions are randomly distributed initially. After five generations, solutions start converging toward the promising region for locating the optima. After 10 generations, solutions are converged near to the optimal solution. Finally, all solutions get converged in 25 generations. Figure 4.2d shows that the optimal solution achieved the desired target reliability.

Table 4.3: RBDO results for Example 2 with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC	β^t_{MCS}
			G_{FC}	G_1
DLM-PMA	-	-	-	-
CC($\lambda = 0.2$)	37.3930	(3.5744, 3.7625)	3751	3.0814
SORA	36.7965	(2.9152, 3.9933)	2176	2.8109
ASLCC	37.3927	(3.6210, 3.7245)	333	3.0843
SLA	-	-	-	-
DLRBDO-DE	37.3167	(3.5780, 3.7443)	640888	3.0459
SLADE (Proposed)	37.3835	(3.5038, 3.8175)	25500	3.0701

Table 4.4: Statistical values obtained by SLADE for Example 2

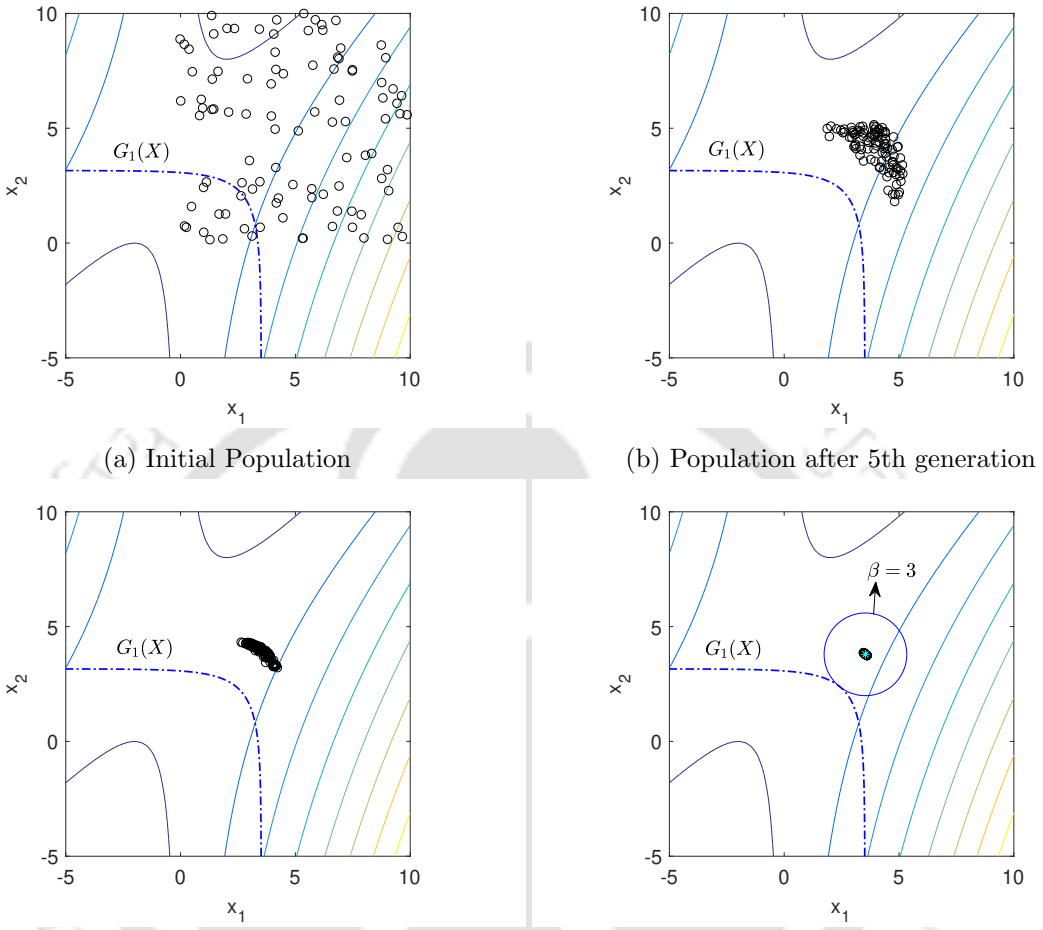
Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	37.3845	37.3835	37.3836	0.0011

4.3.2 Example 2

The second RBDO example consists of a non-linear performance function, which is concave in nature with a non-linear objective function. The formulation for RBDO example is given in equation (3.16) of section 3.3.2.

Table 4.3 presents the obtained solutions from different methods. It can be seen from the fifth column of the table that SLADE, DLRBDO-DE, CC, and ASLCC generate the reliable solutions for this example. Among them, DLRBDO-DE generates the best reliable solution followed by SLADE based on the objective function value. It can be seen that SLA and DLM-PMA are unable to converge for this example. This is primarily due to computation of steepest descent search for estimating MPTP. Steepest descent works well for convex function but found unstable while handling concave function. SORA generates a solution for the given RBDO example but fails to achieve the desired target reliability because it also estimates MPTP by steepest descent direction.

The fourth column of Table 4.3 reveals that SLADE needs more function evaluations than CC and SORA. However, SLADE needs almost 25 times less function evaluations than DLRBDO-DE. The statistical values obtained by SLADE are presented in Table 4.4. It can be observed that the standard deviation in objective function values for running SLADE 50 times with different initial population is 1.1×10^{-3}, which again demonstrates consistency of the proposed meta-heuristic algorithm. Similar to the previous example, Figure 4.3 shows progress of solutions of SLADE in different generations. It can be observed that solutions



(a) Initial Population

(b) Population after 5th generation

(c) Population after 10th generation

(d) Population after 25th generation

Figure 4.3: Progress of solutions in different generations for Example 2.

were randomly distributed initially and then, converged to the optimal solution generation by generation. Figure 4.3d shows that the desired target reliability is achieved by solutions for this example.

4.3.3 Example 3

The RBDO model of example 3 (Deb et al., 2009) consists of a one non-linear and two linear performance functions with a linear objective function that is given in equation (4.8). There

Table 4.5: RBDO results for Example 3 with $\beta^t = 4.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC	β^t_{MCS}
			G_{FC}	G_1, G_2, G_3
DLM-PMA	-27.5574	$(115.8741, -27.5574)$	2079	$4.0128, 4.0128, \text{Inf}$
CC($\lambda = 0.2$)	-27.4904	$(115.9971, -27.4904)$	25452	$4.0175, 3.9444, \text{Inf}$
SORA	-27.5571	$(115.8741, -27.5571)$	711	$4.0125, 3.8909, \text{Inf}$
SLA	-27.5574	$(115.8741, -27.5574)$	302	$3.9444, 4.0128, \text{Inf}$
FastRIA+NSGA-II[1]	11.8200	$(-237.908, 11.820)$	–	$4.1075, \text{Inf}, 4.0128$
DLRBDO-DE	–	–	–	–
SLADE (Proposed)	12.1812	$(-236.9653, 12.1812)$	76500	$4.0962, \text{Inf}, 4.0208$

1 Deb et al. (2009)

Table 4.6: Statistical values obtained by SLADE for Example 3

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	12.1811	12.1812	12.1812	0.0019×10^{-2}

are two independent random normal variables, each with standard deviation of 10. The desired target reliability is $\beta^t = 4.0$ for all performance functions. The initial design point for classical optimization RBDO methods are taken as $\boldsymbol{\mu}_x^{(0)} = [100, 100]^T$.

$$
\begin{aligned}
&\text{max.: } \mu_{x_2}, \\
&\text{s.t.: } P\left[G_i(\mathbf{X}) \geq 0\right] \leq \phi(-\beta_i^t), \quad i = 1, 2, 3, \\
&\quad G_1(\mathbf{X}) = x_1^2 - 1000x_2, \\
&\quad G_2(\mathbf{X}) = x_2 - x_1 + 200, \\
&\quad G_3(\mathbf{X}) = x_1 - 3x_2 + 400, \\
&\quad -400 \leq \boldsymbol{\mu}_{x_l} \leq 300, \; x_l \sim N(\boldsymbol{\mu}_{x_l}, 10^2) \text{ for } l = 1, 2, \\
&\quad \beta_i^t = 4.0, \boldsymbol{\mu}_x^{(0)} = [100, 100]^T.
\end{aligned} \tag{4.8}
$$

Table 4.5 presents solutions obtained by various methods. It can be seen from the second column of the table that SLADE and FastRIA+NSGA-II are able to generate a solution with maximum objective function. The solution is also found reliable that can be seen from the fifth column of the table. Apart from SLADE and FastRIA+NSGA-II, DLM-PMA is able to generate reliable solution. However, the solution evolved by DLM-PMA is inferior. Other methods like CC, SORA, and SLA are unable to generate the reliable solution for this example. DLRBDO-DE is also unable to converge because the convergence of MPTP using PMA gets

Table 4.7: RBDO results for speed reducer example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC
			G_{FC}
DLM-PMA	3038.6125	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	229680
CC($\lambda = 0.2$)	3038.6125	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	2439888
SORA	3038.6125	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	14874
SLA	3038.6125	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	1038
SSRBO[1]	3038.5300	(3.576, 0.7000, 17.0000, 7.3000, 7.754, 3.366, 5.301)	–
LAS[2]	3038.624	(3.5765, 0.7000, 17.0000, 7.3000, 7.7543, 3.3651, 5.3017)	–
DLRBDO-DE	3038.7232	(3.5766, 0.7000, 17.0000, 7.3037, 7.7572, 3.3650, 5.3017)	142746780
SLADE (Proposed)	3038.6125	(3.5765, 0.7000, 17.0000, 7.3000, 7.7541, 3.3652, 5.3017)	5832750

	β_{G_1}	β_{G_2}	β_{G_3}	β_{G_4}	β_{G_5}	β_{G_6}	β_{G_7}	β_{G_8}	β_{G_9}	$\beta_{G_{10}}$	$\beta_{G_{11}}$
SSRBO	Inf	Inf	Inf	Inf	3.1501	2.8671	Inf	2.9786	Inf	Inf	3.1039
LAS	Inf	Inf	Inf	Inf	2.9651	3.0260	Inf	3.0057	Inf	Inf	3.0073
DLRBDO-DE	Inf	Inf	Inf	Inf	2.9498	3.0382	Inf	3.0511	Inf	Inf	3.4702
Other methods	Inf	Inf	Inf	Inf	3.0057	3.0052	Inf	3.0002	Inf	Inf	2.9953

1(Li et al., 2019), 2 (Chen et al., 2014)

Table 4.8: Statistical values obtained by SLADE for speed reducer example

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	3038.7231	3038.6125	3038.6143	0.0077

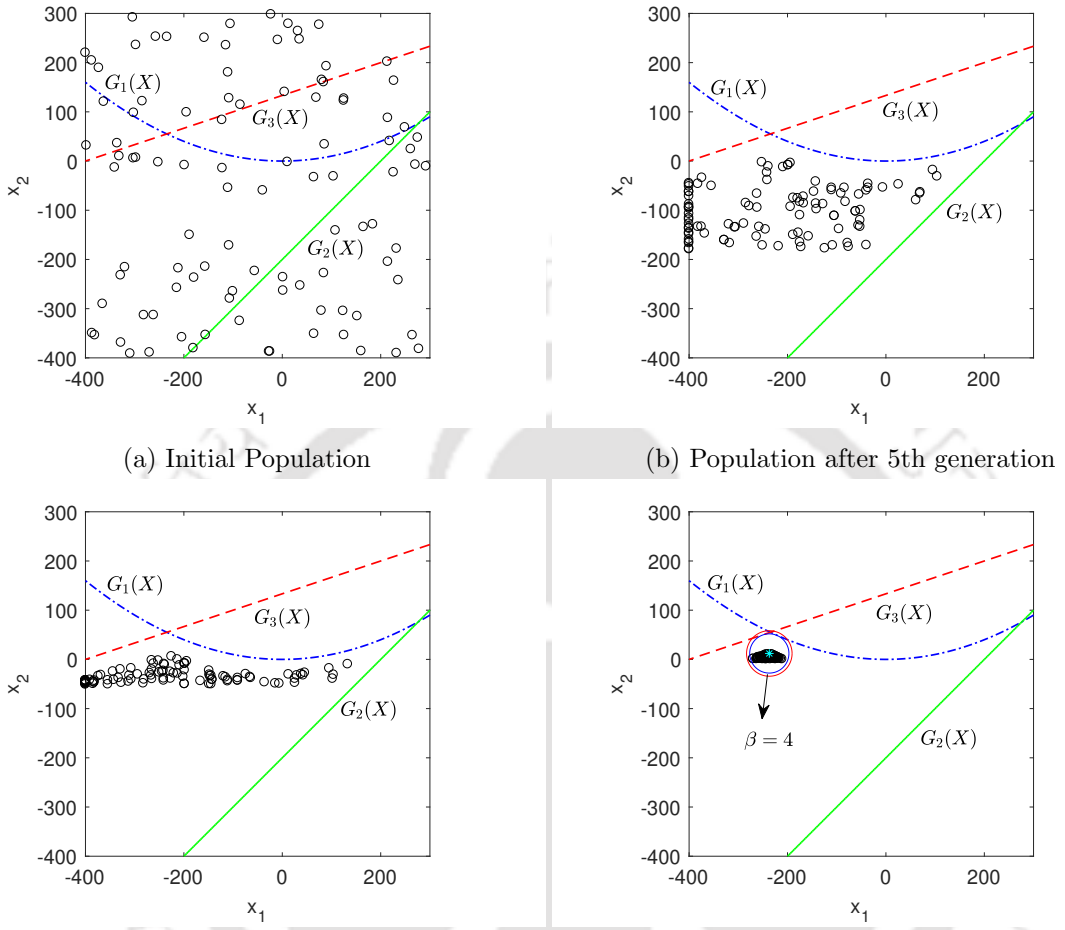
oscillated.

The forth column of Table 4.5 presents the number of function evaluations. It can be seen that SLADE needs many function evaluations for generating the optimal and reliable solution as compared to DLM-PMA. Since other methods cannot generate a reliable solution, the number of function evaluations is not compared.

Table 4.6 presents statistical results obtained by SLADE. For this example again, SLADE is found consistent in generating the solution. It is noted that the scaling factor of DE is considered as 1.2. Figure 4.4 shows the progress of solutions in different generations. Similar to previous examples, solutions that were randomly generated in the variable space get converged to the optimal solution with generations. Figure 4.4d demonstrates that one of the solutions is the optimal and achieves the desired target reliability.

4.3.4 Speed reducer example

The speed reducer problem discussed in section 3.3.5 is considered as the next example. The objective is to reduce the weight of the speed reducer, which is subjected to performance



(a) Initial Population

(b) Population after 5th generation

(c) Population after 10th generation

(d) Population after 25th generation

Figure 4.4: Progress of solutions in different generations for Example 3.

functions. The RBDO problem formulation is given in equation (3.19).

Table 4.7 presents the RBDO results of different methods. It can be seen that all the methods converge to the same solution, except for DLRBDO-DE, SSRBDO and LAS. The number of function evaluation shown in fourth column of the table measures the efficiency of the methods. It can be seen that SLADE required more function evaluation than DLM-PMA, CC, SORA and SLA. However, it is 24× more efficient than DLRBDO-DE. These methods generate the solution with the same target reliability, except for DLRBDO-DE, SSRBDO and LAS. It can be also be observed that none of the methods generate the solution achieving the target reliability for all the performance functions. However, the target reliability obtained by

Table 4.9: RBDO results for spring design example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC	β^t_{MCS}
			G_{FC}	G_1, G_2, G_3
DLM-PMA	2.3142×10^{-2}	$(0.0589, 0.4620, 12.4319)$	24480	2.9803, 3.0091,Inf,Inf
CC($\lambda = 0.2$)	2.3142×10^{-2}	$(0.0589, 0.4621, 12.4286))$	28744	2.9760, 3.0000,Inf,Inf
SORA	2.3120×10^{-2}	$(0.0593, 0.4730, 11.9008)$	1979	2.9781, 3.0091,Inf,Inf
SLA	2.3133×10^{-2}	$(0.0586, 0.4545, 12.7981)$	653	3.0000, 2.9576,Inf,Inf
SLSVCG	2.3142×10^{-2}	$(0.0589, 0.4607, 12.4970)$	1281	2.9867, 3.0433,Inf,Inf
DLRBDO-DE	-	-	-	-
SLADE (Proposed)	2.3116×10^{-2}	$(0.0590, 0.4650, 12.2815)$	424200	3.0138, 2.9911,Inf,Inf

Table 4.10: Statistical values obtained by SLADE for spring design example

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	2.3265×10^{-2}	2.3116×10^{-2}	2.3148×10^{-2}	0.1689×10^{-5}

SLADE is better than DLRBDO-DE, SSRBDO and LAS. The statistical values of objective function obtained by SLADE are presented in Table 4.8. Again, SLADE shows its consistent performance for solving this example.

4.3.5 Spring design example

The spring design example is considered as the next example and the RBDO formulation is given in equation (3.20) of section 3.3.6.

Table 4.9 presents the RBDO results obtained from different methods. It can be observed that none of the methods are able to obtain the target reliability for all the performance functions. However, β^t obtained by SLADE is better than rest of the methods. The solution obtained by SLSVCG that uses conjugate gradient direction to estimate MPTP is also unable to achieve the target reliability index. The target reliability index obtained by SLA is poor compared to rest of the methods. The objective function value obtained by SLADE is also found to be better than other methods. For this example also, DLRBDO-DE is unable to converge. Comparing NFC, SLADE required most function evaluation than other methods because it is a population-based meta-heuristic algorithm. The statistical values of different runs of SLADE are presented in Table 4.10, which again shows its consistency in generating the solution for spring design example.

Table 4.11: RBDO results for welded beam example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}^*_{\mathbf{X}}$	NFC	β^t_{MCS}
			G_{FC}	G_1, G_2, G_3
DLM-PMA	2.5913	(5.73, 200.8982, 210.5977, 6.2389)	57750	3.0233, 3.0233, 3.0091,Inf,3.0091
CC($\lambda = 0.2$)	2.5910	(5.7306, 200.8333, 210.5977, 6.2389))	660550	3.0185, 3.0433, 3.0258Inf,3.0564
SORA	2.5913	(5.73, 200.8982, 210.5977, 6.2389)	2155	3.0233, 3.0233, 3.0091,Inf,3.0091
SLA	2.5913	(5.73, 200.8982, 210.5977, 6.2389)	847	3.0233, 3.0233, 3.0091,Inf,3.0091
ASORA	2.5913	(5.73, 200.8982, 210.5977, 6.2389)	905	3.0233, 3.0233, 3.0091,Inf,3.0091
DLRBDO-DE	2.5070	(5.6644, 188.8490, 211.6497, 6.2029)	7451958	1.3542,Inf,3.1786,Inf,0.3029
SLADE (Proposed)	2.5915	(5.7284, 200.5421, 211.0867, 6.2374)	454500	3.0233,Inf,3.0045,Inf,3.1865

Table 4.12: Statistical values obtained by SLADE for welded beam example

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	2.9609	2.5915	2.6216	0.0740

4.3.6 Welded beam example

The welded beam example consists of four random variables. These random variables are statically independent and follows normal distribution. The objective function is to minimize the welding cost. The RBDO formulation of the problem is given in equation (3.21) of section 3.3.7.

Table 4.11 presents the results obtained from all methods. It can be observed that all the methods are able to obtain the target reliability for all performance functions, except for DLRBDO-DE. The target reliability index obtained by SLADE is better than other methods. Methods, like DLM-PMA, SORA, SLA, and ASORA converge to the same optimal solution and thus, have the same β^t. Comparing NFC, SLADE is 16$\times$ more efficient than DLRBDO-DE. The objective function value obtained by CC method is better than SLADE but with an expense of extra computational cost. Table 4.12 presents the statistical values of objective function obtained by SLADE, which shows consistency of the method in generating a reliable solution for the given example.

4.3.7 Cantilever beam example

A cantilever beam loaded with two point loads, i.e., lateral load p_z and vertical load p_y at the tip is considered as the next example. The objective function is to minimize the weight of the beam and there are two performance functions related to maximum stress and displacement. The RBDO formulation is presented in equation (3.22) of section 3.3.8.

Table 4.13 presents the results obtained from different methods. It can be seen that all

Table 4.13: RBDO results for cantilever beam example with $\beta^t = 3.0$

Methods	f^*	$\boldsymbol{\mu}_{\mathbf{X}}^*$	NFC	β_{MCS}^t
			G_{FC}	G_1, G_2, G_3
DLM-PMA	9.5253	(2.4538, 3.8819)	13930	3.0025, 3.0320
CC($\lambda = 0.2$)	9.5230	(2.4537, 3.8809)	113386	3.0233, 2.9719
SORA	9.2840	(2.5802, 3.5981)	2498	2.5696, 3.0000
SLA	9.5253	(2.4538, 3.8819)	370	3.0025, 3.0320
SLSV-CG	9.5253	(2.4538, 3.8819)	384	3.0025, 3.0320
ASORA	9.5253	(2.4538, 3.8819)	570	3.0025, 3.0320
IRA-DE[1]	9.5202	(2.4488, 3.8877)	–	3.0152, 3.0720
DLRBDO-DE	-	-	-	-
SLADE (Proposed)	9.5153	(2.4210, 3.9301)	262600	3.4768, 3.0727

1 (Lobato et al., 2017)

Table 4.14: Statistical values obtained by SLADE for cantilever beam example

Method	Worst Obj.	Best Obj.	Mean Obj.	Standard Deviation
SLADE (Proposed)	9.7448	9.5153	9.5470	0.0847

the methods achieve the target reliability, except for CC and SORA. The objective function value obtained from SLADE is better than rest of the methods. It can also be seen that β^t obtained from SLADE is also better than other methods. For this example also, DLRBDO-DE is unable to converge. The main reason of a failure of DLRBDO-DE is owing to oscillation in convergence of MPTPs by steepest descent direction. The statistical data of SLADE obtained from different runs is presented in Table 4.14. For this example also, SLADE shows consistent performance in generating the best reliable solution.

4.4 Closure

In the last section, seven examples including three mathematical and four engineering RBDO examples were solved to demonstrate the efficacy and consistency of SLADE. The first two examples consist of nonlinear convex and concave performance functions. It was observed that the solution generated by SLADE satisfied the desired reliability with the best objective function value. The third mathematical example has local and global solutions. Other variant of DE, that is, DLRBDO-DE was unable to converge to near optimal solution for example 3, spring design and cantilever beam examples. It was observed that the convergence of reliability analysis performed using PMA got stuck in a periodic oscillation for highly nonlinear perfor-

mance function. SLADE was able to obtain reliable solutions with better objective function values for five examples. Since SLADE is a population based meta-heuristic algorithm, it consumes more function evaluation than RBDO methods developed using numerical optimization techniques. However, the number of function evaluations obtained by SLADE is found quite less than DLRBDO-DE.

Chapter 5

Single-loop multi-objective RBDO methods

Chapter 4 presented a novel single-loop method for obtaining global reliable solution. However, the method is limited to single objectives. On the other hand, most of the real-world problems consist of multiple objectives apart from uncertainty. The major challenge is to obtain reliable Pareto optimal solutions for multi-objective problems. Therefore, a single-loop based multi-objective formulation for RBDO is proposed. Since concepts like shifting vector approach and chaos control theory are already explored in the previous chapters, the same are adopted with the proposed formulation. The formulation is solved using multi-objective DE that is designed using non-dominated sorting and crowding distance. The algorithm is made adaptive for better convergence by introducing a heuristic parameter for generating mutant vectors from different variants of DE as mentioned earlier. The heuristic parameter is estimated in every generation using hypervolume performance indicator. In the following sections, the proposed single-loop multi-objective reliability-based design optimization is discussed.

5.1 Proposed single-loop multi-objective RBDO formulation

The multi-objective formulation presented in equation (2.35) includes reliability analysis that can be handled using PMA or related methods. Using SLA, the proposed multi-objective formulation for RBDO can be written as

$$\begin{array}{lll} \text{Minimize} & f_m(\boldsymbol{\mu}_{\mathbf{X}}), & m = 1, \ldots, M, \\ \text{subject to} & G_i(\mathbf{X}^{*(k)}_{i,SLCC}) \geq 0, & i = 1, \ldots, I, \end{array} \tag{5.1}$$

where $\mathbf{X}_{i,SLCC}^{*(k)}$ is the MPTP calculated for $i-$th constraint or performance function in $k-$th iteration/generation. The value of $\mathbf{X}_{i,SLCC}^{*(k)}$ is calculated using the following transformation

$$\mathbf{X}_{i,SLCC}^{*(k)} = \mathrm{T}^{-1}(\mathbf{U}) = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} + \boldsymbol{\sigma}_{\mathbf{X}}\mathbf{U}_{i,SLCC}^{(k)}. \tag{5.2}$$

Here, $\mathbf{U}_{i,SLCC}^{(k)}$ is the MPTP estimated in the standard normal space using the chaos control theory (Biswas and Sharma, 2021a) that is given as

$$\mathbf{U}_{i,SLCC}^{(k)} = \beta_i^t \frac{\mathbf{U}_i^{(k-1)} + \lambda_i^{(k)}\mathbf{C}[\mathbf{U}_i^{(k)} - \mathbf{U}_i^{(k-1)}]}{\|\mathbf{U}_i^{(k-1)} + \lambda_i^{(k)}\mathbf{C}[\mathbf{U}_i^{(k)} - \mathbf{U}_i^{(k-1)}]\|}. \tag{5.3}$$

In equation (5.3), $\mathbf{U}_i^{(k)}$ is the MPTP calculated for $i-$th constraint in $k-$th generation in the standard normal space. Its value is calculated after transformation that is given as

$$\mathbf{U}_i^{(k)} = \mathrm{T}(\mathbf{X}_i^{(k)}) = (\mathbf{X}_i^{(k)} - \boldsymbol{\mu}_{\mathbf{X}})/\boldsymbol{\sigma}_X. \tag{5.4}$$

The formulation given in equation (5.1) is unique with respect to the literature because it is developed using the single-loop methodology that eliminates the nested double-loop formulation. Moreover, the reliability analysis is approximated by estimating the deterministic constraints through KKT optimality conditions by adapting SLA given in equation (3.6). Furthermore, the MPTP is estimated in the proposed formulation using the chaos control theory.

5.2 Multi-objective differential evolution using adaptive mutation scheme

DE for multi-objective optimization with adaptive mutation scheme is discussed with the help of the following steps.

Step 1 **Input**: Population size (N), number of variables (n), total number of generations (K), scaling factor ($\hat{F}$), probability of crossover (p_c), standard deviation ($\boldsymbol{\sigma}$) for random variables, target reliability index for constraints (β^t), generation counter ($k = 1$), chaos control factor λ and involutary matrix $\mathbf{C} = \mathbf{I}$.

Step 2 Generate random initial population ($P(k)$) consisting of target vectors.

Step 3 Evaluate each target vector ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$), $t \in N$ of ($P(k)$)

Step 3.1 Calculate objective function values, $f_m(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)),\ \forall m \in M$

Step 3.2 Calculate constraint violation of each performance function using the MPTP estimated through chaos control theory given in equation (5.1).

Step 4 Generate mutant vectors ($\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}$) using the proposed adaptive mutation schemes given in equation (5.6).

Step 5 Generate trial vectors ($\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}$) using equation (5.8).

Step 6 Evaluate each trial vector

Step 6.1 Calculate objective function values, $f_m(\boldsymbol{\mu}_{\mathbf{X}}^{(v)}(t)),\ \forall m \in M$

Step 6.2 Calculate constraint violation of each performance function using the MPTP estimated through chaos control theory given in equation (5.1).

Step 7 Combine target and trial vectors and perform non-dominated sorting and estimate crowding distance

Step 8 Select the next generation target vectors ($\boldsymbol{\mu}_{\mathbf{X}}^{(k+1)}$) using the environmental selection of NSGA-II and set $k = k + 1$.

Step 9 If $(k > K)$, terminate and report the non-dominated target vectors. Otherwise, go to Step 4.

In Step 1, various input required for executing DE are given. The list also includes some input parameters for the proposed formulation. In Step 2, initial random population is generated that consists of a set of solutions. These solutions are referred to as target vectors in DE. Since DE is used for solving RBDO problem, the notations are kept the same to represent a target vector. For example, $\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$ represents $t-$th target vector in $k-$th generation. In Step 3, each target vector is evaluated. The objective function values are calculated using the mean value ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$) of the random variable $\mathbf{X}^{(k)}(t)$. Thereafter, constraint violation is calculated for each constraint using the estimated MPTP through chaos control theory using equations (5.1)$-$(5.4).

In Step 4, mutant vectors are calculated ($\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t)$) that is given as

$$\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t) = \boldsymbol{\mu}_{\mathbf{r}_1}^{(k)}(t) + \hat{F} \times (\boldsymbol{\mu}_{\mathbf{r}_2}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r}_3}^{(k)}(t)), \tag{5.5}$$

where $\mathbf{r_1} \neq \mathbf{r_2} \neq \mathbf{r_3}$ are randomly chosen target vectors from the current population, and $\hat{F}$ is the scaling factor. As discussed in section 4.2.2, this DE variant is found effective in exploring the search space that is desired during the initial generations. However, when DE starts converging towards the Parto-optimal front, the DE variant replacing $\boldsymbol{\mu}_{\mathbf{r}_1}^{(k)}(t)$ to $\boldsymbol{\mu}_{\mathbf{best}}^{(k)}(t)$ can be useful since the best vector can guide the algorithm for better convergence. The $\boldsymbol{\mu}_{\mathbf{best}}^{(k)}(t)$ vector for each target vector is found by calculating the Euclidean distance of the $t-$th target vector with respect to all non-dominated target vectors in the objective space. The closest non-dominated target vector is selected as $\boldsymbol{\mu}_{\mathbf{best}}^{(k)}(t)$ for the $t-$th target vector. This procedure is followed for all target vectors in the current generation.

Since both the variants are useful at different stages of progress of DE, a heuristic parameter (ζ) is proposed for adaptively generating mutant vector either using random vector or the best. The adaptive mutation scheme is given as

$$\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t) = \begin{cases} \boldsymbol{\mu}_{\mathbf{r1}}^{(k)}(t) + \hat{F} \times (\boldsymbol{\mu}_{\mathbf{r2}}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r3}}^{(k)}(t)), & \zeta > \epsilon, \\ \boldsymbol{\mu}_{\mathbf{best}}^{(k)}(t) + \hat{F} \times (\boldsymbol{\mu}_{\mathbf{r2}}^{(k)}(t) - \boldsymbol{\mu}_{\mathbf{r3}}^{(k)}(t)), & \text{otherwise}, \end{cases} \tag{5.6}$$

where ϵ is a user-defined parameter. The parameter ζ is calculated using hypervolume performance indicator that is given as

$$\zeta = I_H^{(k)} - I_H^{(k-1)}. \tag{5.7}$$

Here, $I_H^{(k)}$ and $I_H^{(k-1)}$ are the hypervolume calculated with respect to the non-dominated target vectors in (k) and $(k-1)$ generations. $I_H^{(k)}$ measures the hypervolume of that portion of the objective space which is weakly dominated by an approximate set of solutions, and is to be maximized at $k-$th iteration. It is noted that the non-dominated target vectors in $(k-1)$ and (k) generations are normalized together for estimating the hypervolume with respect to the dominated point $(1.1, \ldots, 1.1)^T$.

In Step 5, the trial vectors are generated using the following equation.

$$\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j) = \begin{cases} \boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j) & \text{if } r \leq p_c \text{ or } j = rnbr(i), \\ \boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t_j) & \text{if } r > p_c \text{ and } j \neq rnbr(i), \end{cases} \tag{5.8}$$

where subscript j with t in $\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t_j)$, $\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j)$, and $\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j)$ represents $j-$th component of target, mutant and trial vectors, respectively. r is a random number between 0 to 1, p_c is the crossover rate, and $rnbr(i)$ is a randomly chosen index $\in \{1, 2, \ldots, n\}$, which ensures that $\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t_j)$ gets at least one component from $\boldsymbol{\mu}_{\mathbf{V}}^{(k+1)}(t_j)$.

Once all trial vectors are generated, these vectors are evaluated in Step 6. The procedure for evaluating the objective function values and constraint violation remains the same as discussed in Step 3. Thereafter, in Step 7, the target vectors and trial vectors are combined ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t) \bigcup \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t)$). The rank of the combined population is calculated using non-dominated sorting (Deb et al., 2002) and crowding distance is calculated for diversity. In Step 8, the best N target vectors for the next generation are selected by adapting environmental selection of NSGA-II (Deb et al., 2002). In Step 9, multi-objective DE is terminated if the generation counter (k) is more than the total number of generations (K). Otherwise, algorithm goes back to Step 4 and performs the remaining steps till the termination condition gets satisfied.

5.3 Numerical examples

In this section, the proposed method is tested on two mathematical and one engineering RBDO examples. All the examples are consisting of two objectives. The proposed method that is now referred to as single-loop multi-objective differential evolution (SL-MODE) is compared with a double loop multi-objective differential evolution (DL-MODE). It is noted that PMA is used with DL-MODE for reliability analysis. The reliable Pareto-optimal (PO) solutions are generated by both methods for different values of target reliability index (β^t). Hypervolume (HV) performance indicator and number of function evaluations are used to compare the outcome. Both the methods are run 30 times with different initial populations. The values of the common parameters of both methods are as follows: the scaling factor ($\hat{F}$) is 0.85, the crossover probability (p_c) is 0.8, the population size (N) is 200, and the total number of generations (K) is 100 for the first example, and 200 for second example and 250 for car side impact example. The chaos control factor (λ) is considered as 0.2 (Biswas and Sharma, 2021a). The user defined parameter (ϵ) in equation (5.6) is considered as 10^{-3}. MATLAB R2016b platform is used for developing both methods.

5.3.1 Example 1

A bi-objective RBDO example (Deb et al., 2009) is shown in equation (5.9), which comprises of two independent random normal variables with standard deviation of 0.03. There are two performance functions that are linear.

Table 5.1: Best, median, and worst HV values obtained by both methods are presented for example 1 for different values of β^t. The best performances are highlighted in bold font.

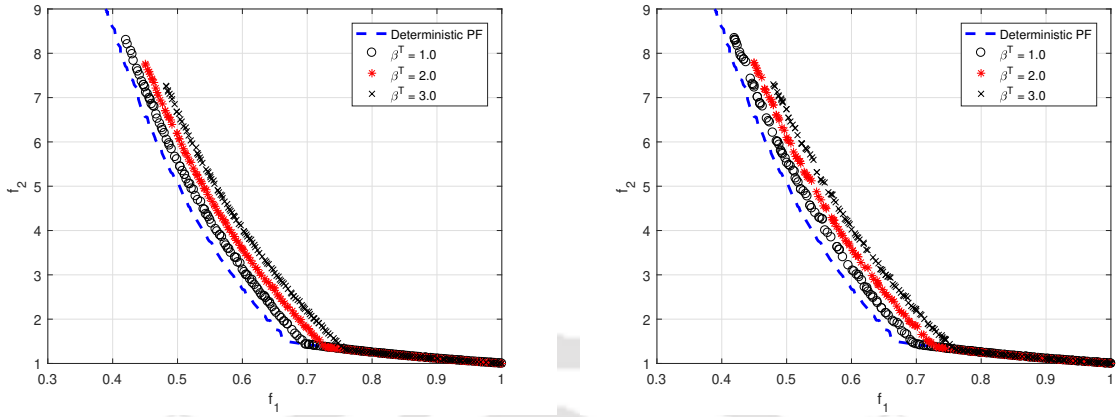
β^t	SL-MODE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE
	0.8076	0.8067		**0.7758**	0.7754		**0.7426**	0.7422
1.0	**0.8071**	0.8048	2.0	**0.7754**	0.7739	3.0	**0.7423**	0.7411
	0.8065	0.7772		**0.7749**	0.7550		**0.7414**	0.7247

$$
\begin{aligned}
&\text{min: } f_1(\boldsymbol{\mu}_{\mathbf{X}}) = \mu_{x_1},\\
&\text{min: } f_2(\boldsymbol{\mu}_{\mathbf{X}}) = \frac{1+\mu_{x_2}}{\mu_{x_1}},\\
&\text{s.t.: } P\left[G_i(\mathbf{X}) > 0\right] \leq \phi(-\beta_i^t), \quad i = 1, 2,\\
&\qquad G_1(\mathbf{X}) = x_2 + 9x_1 - 6,\\
&\qquad G_2(\mathbf{X}) = -x_2 + 9x_1 - 1,\\
&\qquad 0.1 \leq \boldsymbol{\mu}_{x_1} \leq 1,\ 0 \leq \boldsymbol{\mu}_{x_1} \leq 5.
\end{aligned}
\tag{5.9}
$$

Table 5.1 presents the best, median, and worst HV values obtained by both methods. It can be seen from the table that the proposed SL-MODE shows better HV values for different values of β^t. This indicates that the proposed method generates a better set of PO solutions for the given example. It is also important to note that for a larger value of β^t, HV value gets reduced as compared to lower β^t value. It is because larger value of β^t signifies more reliability that makes the obtained PO solutions more conservative and compels them to move away from the deterministic PO front inside the feasible region.

Figure 5.1 shows the obtained PO solutions for different β^t values by both methods. The reliable PO solutions shown in the figure correspond to the run of the median HV value from Table 5.1. The same observation can be seen with both methods that for larger value of β^t the PO solutions become conservative and move away from the deterministic PO front. The same figure also shows that some PO solutions converge to a part of the deterministic PO front that is located at the right bottom. The reason for the same can be found from Figure 5.2 in which the conservative solutions are generated while satisfying the performance function $G_1(\mathbf{x})$. When variable x_2 reaches to its lower limit, the solutions are well positioned inside the feasible region and therefore, both performance functions do not show any effect on them.

The convergence plots of both methods are shown in Figure 5.3. Both methods show smooth convergence of HV values with respect to number of generations. It can be seen that



(a) SL-MODE (b) DL-MODE

Figure 5.1: The obtained PO solutions by both methods for example 1 for different β^t values.

Table 5.2: Number of function evaluations required by both methods for example 1.

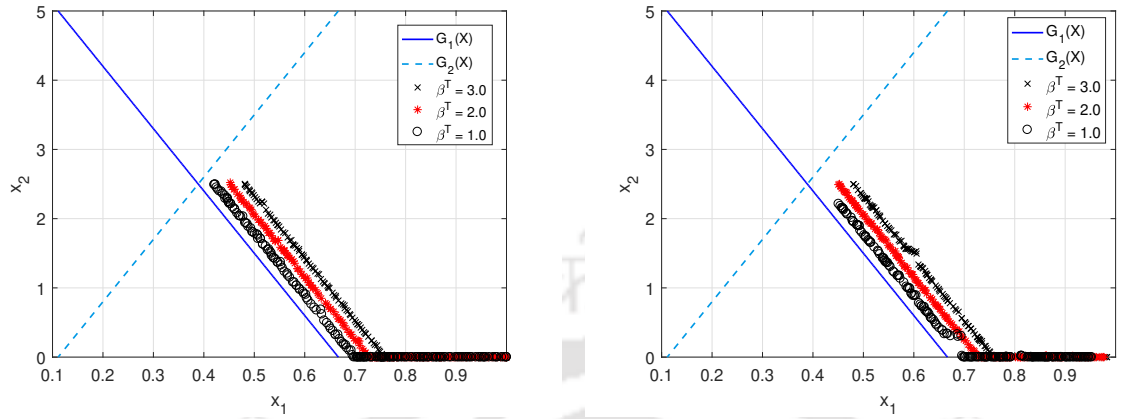
β^t	SL-MODE (Proposed)	DL-MODE
1.0	$2,02,000$	$13,60,000$
2.0	$2,02,000$	$13,60,000$
3.0	$2,02,000$	$13,60,000$

SL-MODE generates the reliable PO solutions in 30 generations for different values of β^t. However, DL-MODE requires 50 generations to get stabilized.

The computational efficacy of both methods is presented in Table 5.2. Number of function evaluations is shown for different β^t values. It can be seen that SL-MODE requires 6.73 times less function evaluations than DL-MODE. It is because SL-MODE satisfies the performance functions using KKT conditions in single-loop approach. However, DL-MODE performs PMA for reliability analysis. Since the population size, number of generations and number of iterations for PMA are kept fixed, the number of function evaluations is the same for different values of β^t.

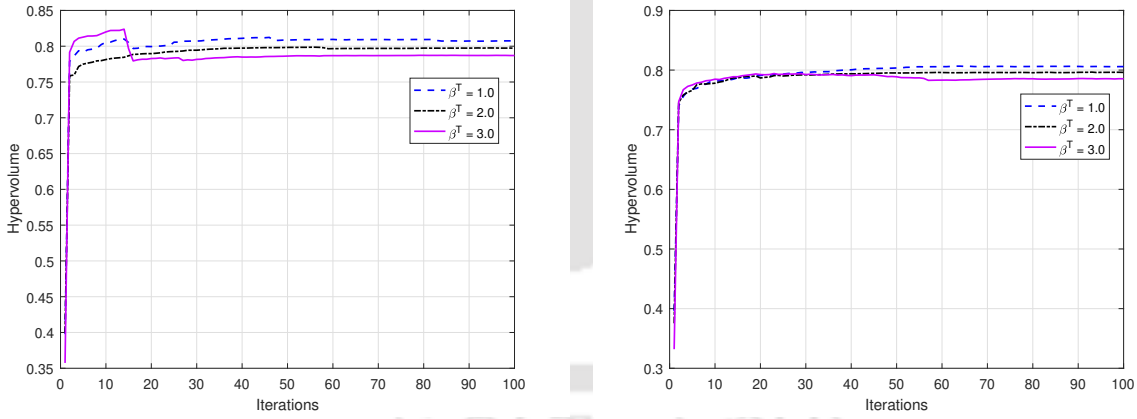
5.3.2 Example 2

The second example (Osyczka and Kundu, 1995) consists of two non-linear objective functions, and four linear and two non-linear performance functions. This example is formulated using two



(a) SL-MODE (b) DL-MODE

Figure 5.2: The obtained PO solutions in the variable space of example 1.



(a) SL-MODE (b) DL-MODE

Figure 5.3: Convergence plots of both methods for example 1 for different β^t values.

independent random normal variables, each with standard deviation of 0.3. The bi-objective

Table 5.3: Best, median, and worst HV values obtained by both methods are presented for example 2 for different values of β^t. The best performances are highlighted in bold font.

β^t	SL-MOE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE
	0.9060	0.8993		**0.6145**	0.6107		**0.3966**	0.3925
1.0	**0.8954**	0.8884	2.0	**0.6061**	0.6015	3.0	**0.3825**	0.3778
	0.8764	0.8279		**0.6006**	0.5944		**0.3721**	0.3678

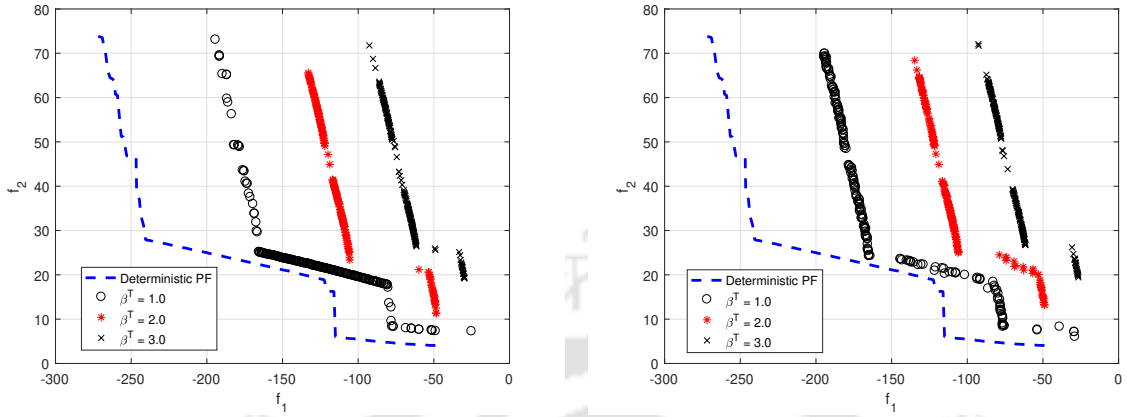
formulation for RBDO example is as follows.

$$
\begin{aligned}
&\text{min: } f_1(\boldsymbol{\mu}_{\mathbf{X}}) = -[25(\mu_{x_1}-2)^2 + (\mu_{x_2}-2)^2 + (\mu_{x_3}-1)^2 + (\mu_{x_4}-4)^2 + (\mu_{x_5}-1)^2],\\
&\text{min: } f_2(\boldsymbol{\mu}_{\mathbf{X}}) = [\mu_{x_1}^2 + \mu_{x_2}^2 + \mu_{x_3}^2 + \mu_{x_4}^2 + \mu_{x_5}^2 + \mu_{x_6}^2],\\
&\text{s.t.: } P\left[G_i(\mathbf{X}) > 0\right] \leq \phi(-\beta_i^t), \quad i = 1, \ldots, 6\\
&\qquad G_1(\mathbf{X}) = x_1 + x_2 - 2, \quad G_2(\mathbf{X}) = 6 - x_1 - x_2,\\
&\qquad G_3(\mathbf{X}) = 2 - x_2 + x_1, \quad G_4(\mathbf{X}) = 2 - x_1 + 3x_2,\\
&\qquad G_5(\mathbf{X}) = 4 - (x_3 - 3)^2 - x_4,\\
&\qquad G_6(\mathbf{X}) = (x_5 - 3)^2 + x_6 - 4,\\
&\qquad 0 \leq \mu_{x_1}, \mu_{x_2}, \mu_{x_6} \leq 10,\ 1 \leq \mu_{x_3}, \mu_{x_5} \leq 5,\ 0 \leq \boldsymbol{\mu}_{x_4} \leq 6.
\end{aligned}
\tag{5.10}
$$

Table 5.3 presents the statistical HV values for different runs of both methods. For different values of β^t, it can be seen that SL-MODE shows the best performance over DL-MODE. In this case also, HV values get reduced for larger value of β^t that is evident from the fact that larger value of β^t compels methods to generate more conservative solutions.

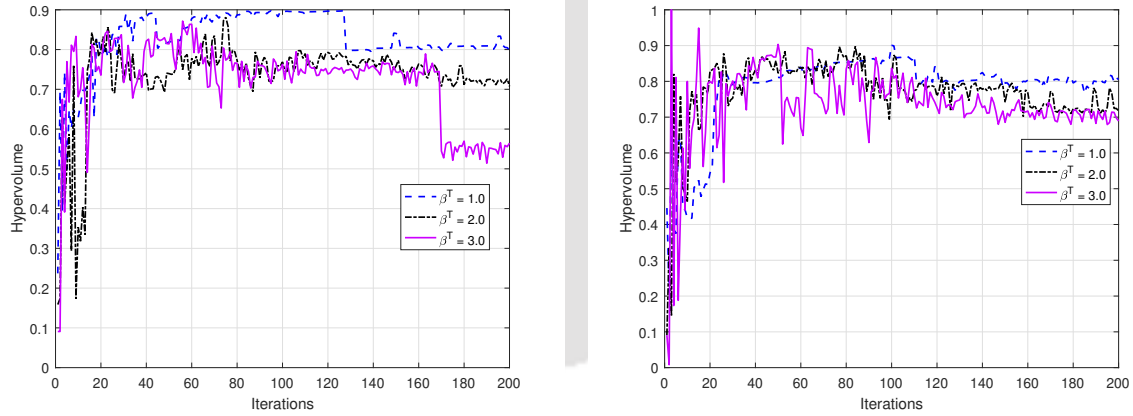
The reliable PO solutions are shown in Figure 5.4 that are generated in the run corresponding to the median HV value in Table 5.3. As β^t value increases, the PO solutions start moving away from the deterministic PO front. It can be seen that the spread of solutions for $\beta^t = 1.0$ is more in case of SL-MODE as compared to DL-MODE. However, the solutions are nicely distributed in case of DL-MODE. As compared to the previous example, the shift of the reliable PO solutions away from the deterministic PO front is more for larger β^t value, resulting into a quite small value of HV than the HV value obtained for smaller β^t. It is also evident from Table 5.3.

The convergence plots of both methods are shown in Figure 5.5. A lot of fluctuations in HV values can be seen with both methods with respect to number of generations. Since the example consists of non-linear objective and performance functions, fluctuations in the convergence are



(a) SL-MODE (b) DL-MODE

Figure 5.4: The obtained PO solutions by both methods for example 2 for different β^t values.



(a) SL-MODE (b) DL-MODE

Figure 5.5: Convergence plots of both methods for example 2 for different β^t values.

observed.

The number of function evaluations required by both methods is presented in Table 5.4. SL-MODE has been determined to be remarkably more efficient than DL-MODE in terms of function evaluation, with a reduction ranging from 26 to 36 times. Since the example is non-linear, PMA requires many function evaluations for reliability analysis.

Table 5.4: Number of function evaluations required by both methods for example 2.

β^t	SL-MODE (Proposed)	DL-MODE
1.0	$31,35,600$	$8,87,43,792$
2.0	$31,35,600$	$10,59,72,384$
3.0	$31,35,600$	$11,47,43,064$

Table 5.5: Details of design variables and their standard deviation values.

Design variable	Standard deviation
x_1: Thickness of B-pillar inner	0.03
x_2: Thickness of B-pillar reinforcement	0.03
x_3: Thickness of floor side inner	0.03
x_4: Thickness of cross members	0.03
x_5: Thickness of door beam	0.03
x_6: Thickness of door beltline reinforcement	0.03
x_7: Thickness of roof rail	0.03
x_8: Material of B-pillar inner	0.006
x_9: Material of floor side inner	0.006
x_{10}: Barrier height	10
x_{11}: Barrier hitting position	10

5.3.3 Car side impact example

The car side impact (Deb et al., 2009) is an engineering RBDO example that is formulated using two objectives and 10 performance functions. There are 11 design variables that are normally distributed and are grouped into random variables $(x_1, \ldots, x_7)$ and random parameters $(x_8, \ldots, x_{11})$. The details of the design variables with their standard deviation values are given in Table 5.5. The bi-objective formulation is given in equation (5.11). The mathematical expressions for each function are given in Table 5.6.

Table 5.6: The objectives and performance functions of car side impact example.

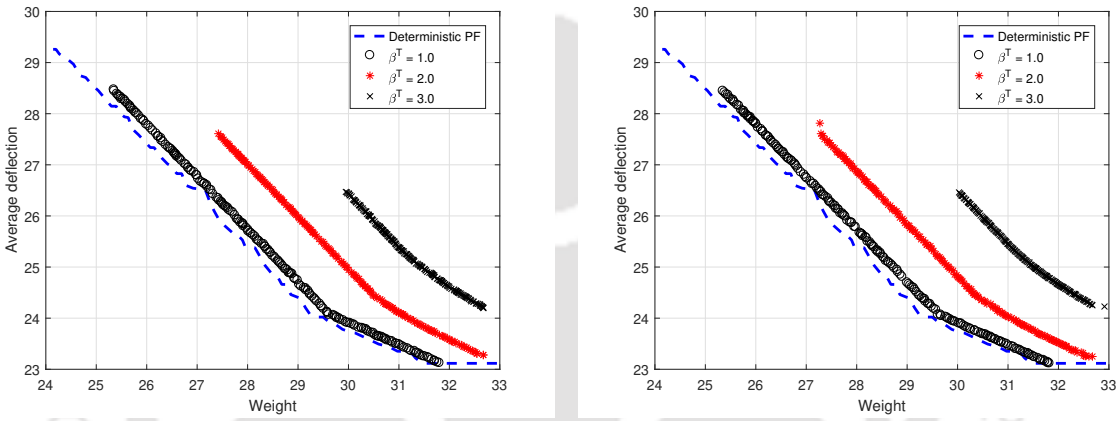
$f_1(\boldsymbol{\mu}_{\mathbf{X}})$:	$1.98 + 4.9x_1 + 6.67x_2 + 6.98x_3 + 4.01x_4 + 1.78x_5 + 0.00001x_6 + 2.73x_7$,
$f_2(\boldsymbol{\mu}_{\mathbf{X}})$:	$(G_5(\mathbf{X}) + G_6(\mathbf{X}) + G_7(\mathbf{X}))/3$,
$G_1(\mathbf{X})$:	$1.16 - 0.3717x_2x_4 - 0.00931x_2x_{10} - 0.484x_3x_9 + 0.01343x_6x_{10}$,
$G_2(\mathbf{X})$:	$0.261 - 0.01598x_1x_2 - 0.188x_1x_8 - 0.0198x_2x_7 + 0.0144x_3x_5 + 0.0008757x_5x_{10}$ $+0.08045x_6x_9 + 0.00139x_8x_{11} + 0.00001575x10x11$
$G_3(\mathbf{X})$:	$0.214 + 0.00817x_5 - 0.1318x_1x_8 - 0.0704x_1x_9 + 0.030998x_2x_6 - 0.018x_2x_7 + 0.0208x_3x_8$ $+0.121x_3x_9 - 0.00364x_5x_6 + 0.0007715x_5x_{10} - 0.0005354x_6x10 + 0.00121x_8x11$ $+0.00184x_9x_{10} - 0.018x_2^2$
$G_4(\mathbf{X})$:	$0.74 - 0.61x_2 - 0.163x_3x_8 + 0.001232x_3x_{10} - 0.166x_7x_9 + 0.227x_2^2$
$G_5(\mathbf{X})$:	$28.98 + 3.818x_3 - 4.2x_1x_2 + 0.0207x_5x_{10} + 6.63x_6x_9 - 7.77x_7x_8 + 0.32x_9x_{10}$
$G_6(\mathbf{X})$:	$33.86 + 2.95x_3 + 0.1792x_{10} - 5.057x_1x_2 - 11.0x_2x_8 - 0.0215x_5x_{10} - 9.98x_7x_8 + 22x_8x_9$
$G_7(\mathbf{X})$:	$46.36 - 9.9x_2 - 12.98x_1x_8 + 0.1107x_3x_{10}$
$G_8(\mathbf{X})$:	$4.72 - 0.5x_4 - 0.19x_2x_3 - 0.01228x_4x_{10} + 0.009325x_6x10 + 0.000191x_{11}^2$
$G_9(\mathbf{X})$:	$10.58 - 0.674x_1x_2 - 1.958x_2x_8 + 0.02054x_3x_{10} - 0.0198x_4x_{10} + 0.028x_6x_{10}$
$G_{10}(\mathbf{X})$:	$16.45 - 0.489x_3x_7 - 0.843x_5x_6 + 0.0432x_9x_{10} - 0.0556x_9x_{11} - 0.000786x_{11}^2$

$$
\begin{aligned}
&\text{min: } f_1(\boldsymbol{\mu}_{\mathbf{X}}) \equiv \text{Structural weight},\\
&\text{min: } f_2(\boldsymbol{\mu}_{\mathbf{X}}) \equiv \text{Average rib deflection},\\
&\text{s.t.: } P\left[G_i(\mathbf{X}) > 0\right] \le \phi(-\beta_i^t), \quad i = 1, \ldots, 10\\
&\quad G_1(\mathbf{X}) = \text{Abdomen load} \le 1\text{KN},\\
&\quad G_2(\mathbf{X}) = V \times Cupper \le 0.32\text{m/s},\\
&\quad G_3(\mathbf{X}) = V \times Cmiddle \le 0.32\text{m/s},\\
&\quad G_4(\mathbf{X}) = V \times Clower \le 0.32\text{m/s},\\
&\quad G_5(\mathbf{X}) = \text{Upper rib deflection} \le 32\text{mm},\\
&\quad G_6(\mathbf{X}) = \text{Middle rib deflection} \le 32\text{mm},\\
&\quad G_7(\mathbf{X}) = \text{Lower rib deflection} \le 32\text{mm},\\
&\quad G_8(\mathbf{X}) = \text{Pubic force} \le 4\text{KN},\\
&\quad G_9(\mathbf{X}) = \text{Velocity of V-Pillar} \le 9.9\text{mm/ms},\\
&\quad G_{10}(\mathbf{X}) = \text{Front door velocity of V-Pillar} \le 15.7\text{mm/ms},\\
&\quad 0.5 \le \mu_{x_1}, \mu_{x_3}, \mu_{x_4} \le 1.5,\ 0.45 \le \mu_{x_2} \le 1.35,\ 0.875 \le \mu_{x_5} \le 2.625,\\
&\quad 0.4 \le \mu_{x_6} \le 1.2,\ 0.4 \le \boldsymbol{\mu}_{x_7} \le 1.2, 0.192 \le \boldsymbol{\mu}_{x_8}, \boldsymbol{\mu}_{x_9} \le 0.75.
\end{aligned}
\tag{5.11}
$$

Table 5.7 presents the statistical HV values obtained after running both methods 30 times with different initial populations. For $\beta^t = 1.0$, DL-MODE shows better HV values. For other

Table 5.7: Best, median, and worst HV values obtained by both methods are presented for car side impact example for different values of β^t. The best performances are highlighted in bold font.

β^t	SL-MODE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE	β^t	SL-MODE (Proposed)	DL-MODE
	0.8156	**0.8216**		**0.7173**	0.7104		**0.5208**	0.5200
1.0	0.8145	**0.8211**	2.0	**0.7137**	0.7095	3.0	**0.5145**	0.5142
	0.8135	**0.8196**		**0.7088**	0.7070		**0.4996**	0.4912



(a) SL-MODE (b) DL-MODE

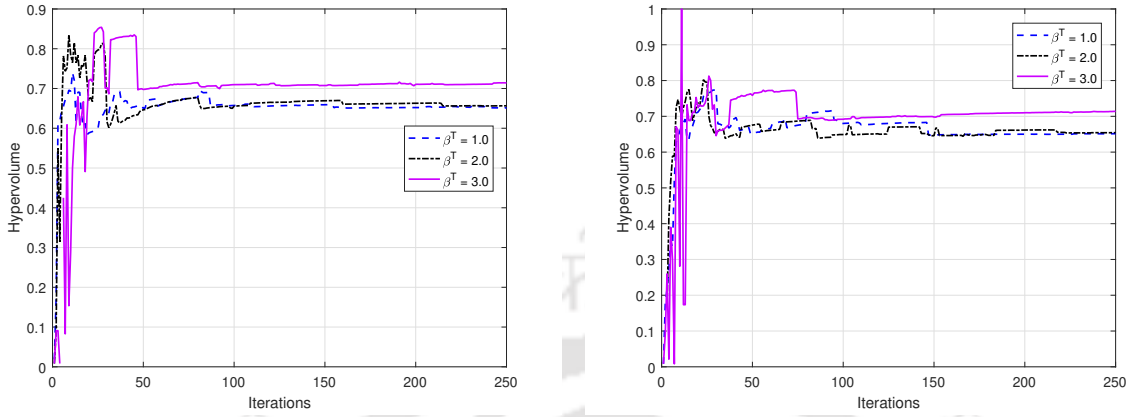
Figure 5.6: The obtained PO solutions by both methods for car impact example for different β^t values.

β^t values, SL-MODE is found better. The observation of reducing HV values with larger β^t value remains the same.

The obtained reliable PO solutions generated by both methods are shown in Figure 5.6. As observed in previous examples, the PO solutions start moving away from the deterministic PO front for larger value of β^t.

The convergence of both methods is shown in Figure 5.7. It can be seen that HV value fluctuates initially for SL-MODE that gets stabilizes after 75 generations for different values of β^t. However, the HV values keep on fluctuating for DL-MODE, except for $\beta^t = 3.0$.

Table 5.8 presents function evaluations required by both methods. Since this example consists of a large number of non-linear functions, SL-MODE showcases a remarkable reduction in function evaluations compared to DL-MODE, with a reduction ranging from 160 to 187 times.



(a) SL-MODE (b) DL-MODE

Figure 5.7: Convergence plots of both methods for car side impact example for different β^t values.

Table 5.8: Number of function evaluations required by both methods for car side impact example.

β^t	SL-MODE (Proposed)	DL-MODE
1.0	$1,15,46,000$	$1.8551e+09$
2.0	$1,15,46,000$	$2.1231e+09$
3.0	$1,15,46,000$	$2.1689e+09$

5.4 Discussion

In the previous section, three examples including two mathematical and one engineering RBDO examples were solved to demonstrate the performance of the proposed method. The first two examples consist of bi-objectives with nonlinear performance functions. The third example was an engineering problem also consists of two objectives and ten performance functions. The proposed SL-MODE and DL-MODE were able to achieve the PO solutions. It was observed that SL-MODE was able to generate better spread of PO solutions.

Both the methods were tested with different values of β^t. It was found that SL-MODE was able to generate better HV values compared to DL-MODE for all the examples. From the figures comparing the convergence plot, it was observed that SL-MODE was was able to converge with less fluctuations. SL-MODE was also found to be computationally efficient

compared to DL-MODE when comparing the number of function evaluation. In example 2, some amount of fluctuation was observed in the convergence plot for both SL-MODE and DL-MODE. This leads to the last objective of the thesis in which shifting vector is incorporated with single-loop multi-objective reliability-based design optimization. In the next section this method is discussed in detail.

5.5 Single-loop MORBDO formulation using chaos control and shifting vector approach

It has been observed from the previous chapters that the concepts of shifting vector and chaos control theory were effective with single-loop formulation. Therefore, these concepts are coupled to develop a new single-loop multi-objective RBDO formulation. By incorporating these changes, the trial vector and target vector become the part of RBDO formulation similar to equation (4.4).

The proposed formulation is given as

$$
\begin{aligned}
&\text{Min. } f_m(\boldsymbol{\mu}_{\mathbf{X}}), && i = 1, \ldots, M, \\
&\text{s.t.: } G_i(\boldsymbol{\Psi}^{(k)}) \geq 0, && i = 1, \ldots, I, \\
&\text{where } \boldsymbol{\Psi}^{(k)} = \begin{cases} \mathbf{X}_{i,MPTP}^{(k)}, & \forall \text{ target vectors}, \\ \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \mathbf{S}_i^{(k+1)}, & \forall \text{ trial vectors}, \end{cases} \\
&\quad \mathbf{S}_i^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \mathbf{X}_{i,MPTP}^{(k)}, \\
&\quad \mathbf{X}_{i,MPTP}^{(k)} = \mathrm{T}^{-1}(\mathbf{U}) = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} + \boldsymbol{\sigma}_{\mathbf{X}} \mathbf{U}_{i,SLCC}^{(k)}, \\
&\quad \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{L})} \leq \boldsymbol{\mu}_{\mathbf{X}} \leq \boldsymbol{\mu}_{\mathbf{X}}^{(\mathrm{U})},
\end{aligned}
\tag{5.12}
$$

where $\mathbf{U}_{i,SLCC}^{(k)}$ is the approximate MPTP in the U-space that is estimated by modified chaos control (Meng et al., 2015) method. $\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}$ is the trial vector in the U-space in $(k+1)$-th iteration. In the proposed formulation, performance function $G_i(\boldsymbol{\Psi}^{(k)})$ includes both $\mathbf{X}_{i,MPTP}^{(k)}$ and $(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \mathbf{S}_i^{(k+1)})$, which are used for evaluating the performance function for each target vector and trial vector, respectively. The vector $(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \mathbf{S}_i^{(k+1)})$ shifts the violated performance function towards the feasible direction for the population of trial vectors. $\mathbf{U}_{i,SLCC}^{(k)}$ in the standard normal space is given in equation (5.3). The value of $\mathbf{U}_{i,SLCC}^{(k)}$ is calculated after the transformation as given in equation (5.4).

5.5.1 Steps for multi-objective DE

The steps of adaptive multi-objective DE remain the same as discussed in section 5.2. Only steps 3 and 6 are modified as per the following details.

- Step [3] For each target vector ($\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t)$) of ($P(k)$)

  3.1 Calculate objective function values, $f_m(\boldsymbol{\mu}_{\mathbf{X}}^{(k)}(t))$.

  3.2 Calculate MPTP for each performance function (i) using equation (5.12) and equation (5.3), and estimate shifting vector $\mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{X}}}^{(k+1)} = \boldsymbol{\mu}_{\mathbf{X}}^{(k)} - \mathbf{X}_{i,MPTP}^{(k)}$.

  3.3 Calculate constraint violation of each performance function using the MPTP that is estimated through chaos control theory given in equation (5.12).

- Step [6] For each trail vector

  6.1 Calculate objective function, $f_m(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)}(t))$.

  6.2 Calculate MPTP ($\hat{\mathbf{X}}_{i,MPTP}^{(k+1)}$) and shifting vector for each performance function (i) using equation (5.12) and equation (5.3), and $\mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{U}}}^{(k+2)} = \boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \hat{\mathbf{X}}_{i,MPTP}^{(k+1)}$ and estimate constraint violation of $G_i(\boldsymbol{\mu}_{\mathbf{U}}^{(k+1)} - \mathbf{S}_{i,\boldsymbol{\mu}_{\mathbf{X}}}^{(k+1)})$.

  6.3 Calculate constraint violation of each performance function using the MPTP that is estimated through chaos control theory given in equation (5.12).

5.6 Numerical Examples

In this section, the same two mathematical examples and one engineering example are solved that are described in section 5.3. The proposed method is abbreviated as SLMDE. The results of SLMDE are compared with a double-loop multi-objective differential evolution (DLMDE). It is noted that PMA is used with DLMDE for reliability analysis. The reliable PO solutions are generated by both methods for different values of target reliability index (β^t). HV performance indicator values and number of function evaluations are used to compare the outcome. Both the methods are run 30 times with different initial populations. The parameters of SLMDE and DLMDE are as follows: the scaling factor ($\hat{F}$) is taken as 0.3, the crossover probability (p_c) is 0.9, the population size (N) is 200, and the total number of generations (K) is 100 for the first example, and 250 for second example and 200 for car side impact example. The chaos control factor (λ) is considered as 0.2 (Biswas and Sharma, 2021a). The user defined parameter

Table 5.9: Best, median, and worst HV values obtained by both methods are presented for example 1 for different values of β^t. The best performances are highlighted in bold font.

β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE
	0.8072	0.8067		**0.7755**	0.7754		**0.7424**	0.7422
1.0	**0.8066**	0.8048	2.0	**0.7752**	0.7739	3.0	**0.7419**	0.7411
	0.8062	0.7772		**0.7746**	0.7550		**0.7412**	0.7247

Table 5.10: Number of function evaluations required by both methods for example 1.

β^t	SLMDE (Proposed)	DLMDE
1.0	$2,02,000$	$13,60,000$
2.0	$2,02,000$	$13,60,000$
3.0	$2,02,000$	$13,60,000$

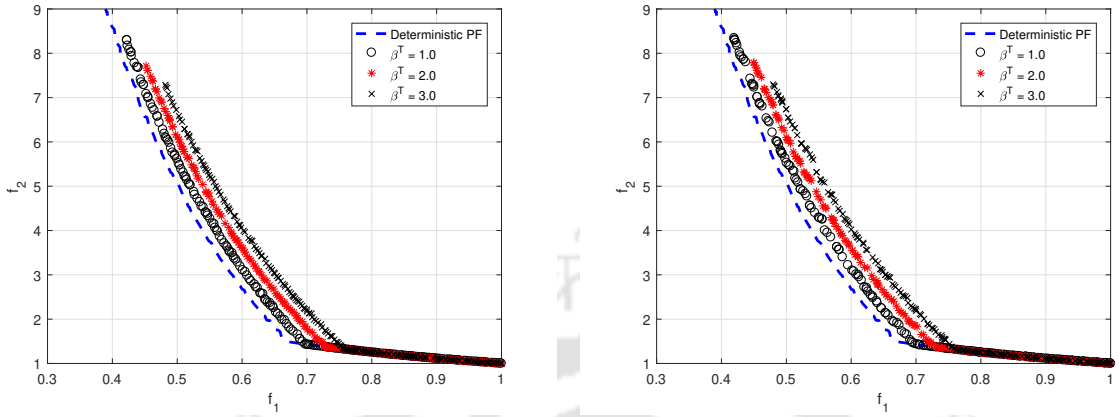
(ϵ) in equation (5.6) is considered as 10^{-3}. MATLAB R2016b platform is used for developing both methods.

5.6.1 Example 1

The first MORBDO example (Deb et al., 2009) consists of two objectives that are developed using two independent random normal variables with standard deviation of 0.03. The example is subjected to two linear performance functions that is shown in equation (5.9) of section 5.3.1.

Table 5.9 presents the best, median, and worst values of HV obtained by SLMDE and DLMDE. SLMDE has converged to better values of HV for different β^t values. This indicates that SLMDE generates a better set of PO solutions for the given example. It is to be noted that for a larger value of β^t, HV value gets reduced as compared to lower β^t value. This is because larger value of β^t signifies high reliability that makes the obtained PO solutions more conservative and push them away from the deterministic PO front inside the feasible region.

Figure 5.8 demonstrates the PO solutions obtained by both methods for different values of β^t. The reliable PO solutions shown in the Figure 5.8a and 5.8b correspond to the median HV values from Table 5.9. It can be seen that for larger values of β^t the PO solutions get conservative and move inside the feasible region. The same figure also demonstrates that some solutions coincide with the deterministic PO front that is located at the right bottom. This is because for those solutions the target reliability is satisfied for the performance function $G_1(\mathbf{x})$.



(a) SLMDE (b) DLMDE

Figure 5.8: The obtained PO solutions by both methods for example 1 for different β^t values.

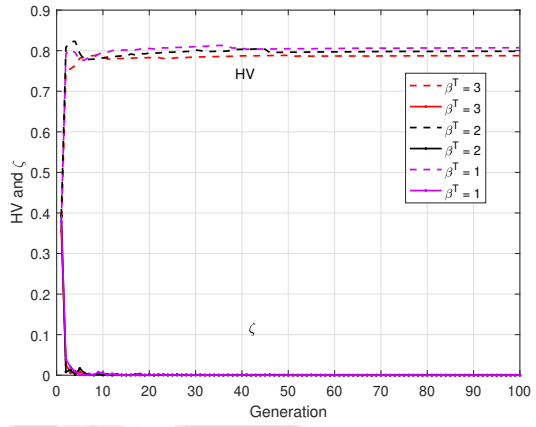


Figure 5.9: Progress of hypervolume and ζ of SLMDE with respect to number of generations for example 1

The computational efficiency of both methods is measured with the help of number of function evaluations that is presented in Table 5.10. It can be seen that the proposed method requires 6.73 times less function evaluations than DLMDE. This is because SLMDE is based on single-loop method where the reliability of performance function is estimated by KKT optimality conditions. On the other hand, DLMDE performs PMA for reliability estimation which requires many function evaluations. Since the number of iterations for PMA is kept fixed, the number of function evaluations is the same for DLMDE with different values of β^t.

The progress of HV and heuristic convergence parameter ζ with respect to iterations is

Table 5.11: Best, median, and worst HV values obtained by both methods are presented for example 2 for different values of β^t. The best performances are highlighted in bold font.

β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE
	0.9118	0.9017		**0.6234**	0.6119		**0.4027**	0.3929
1.0	**0.9028**	0.8906	2.0	**0.6181**	0.6025	3.0	**0.3945**	0.3777
	0.8853	0.8293		**0.5999**	0.5953		**0.3776**	0.3680

Table 5.12: Number of function evaluations required by both methods for example 2.

β^t	SLMDE (Proposed)	DLMDE
1.0	$39,15,600$	$11,14,24,992$
2.0	$39,15,600$	$13,23,44,016$
3.0	$39,15,600$	$14,52,49,728$

shown in Figure 5.9. It can be seen that there are some initial fluctuations in both HV and ζ which subsidise after 10 generations and stabilize after 50 generations.

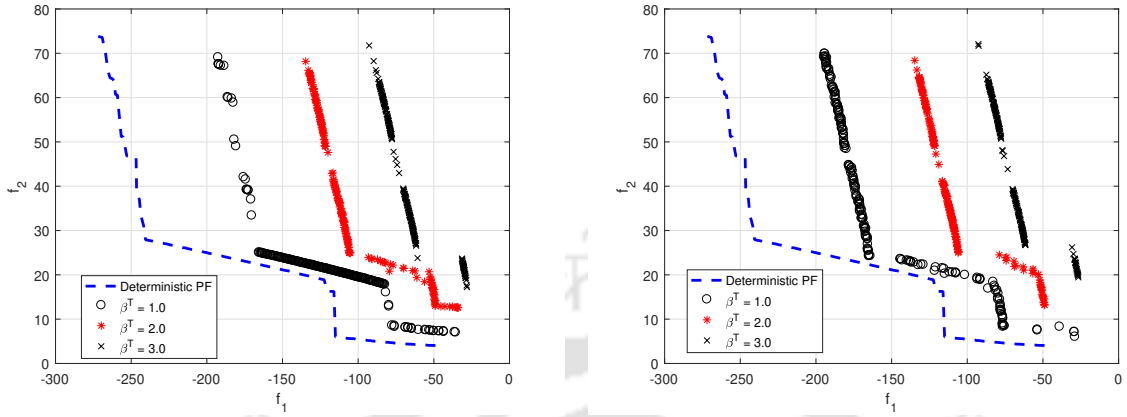
5.6.2 Example 2

The second example consists of two objective functions which are highly non-linear. The RBDO formulation of this example is given in equation (5.10) in section 5.3.2.

Table 5.11 presents the statistical values of HV obtained by both methods. In can be seen that SLMDE has converged to better values of HV for different β^t values. This indicates that SLMDE generates a better set of PO solutions for this given example. In this case also similar observation can be made that for larger values of β^t, the HV values gets reduced.

Figure 5.10 shows the reliable PO solutions generated in the run corresponding to median HV value from Table 5.11. It can be seen that for larger β^t, the PO solutions move inside the feasible region and away from the deterministic PO front. The spread of solutions is less in case of SLMDE for $\beta^t = 1.0$. The solutions are nicely distributed in case of DLMDE. The shift of the solutions is more for larger values of β^t, which leads to smaller values of HV that can be seen from Table 5.11.

Table 5.12 presents the computational efficiency of both methods. The proposed method shows a substantial enhancement in efficiency compared to DLMDE, with a notable range of 28 to 37 times lesser function evaluations required. DLMDE needs many function evaluations



(a) SLMDE (b) DLMDE

Figure 5.10: The obtained PO solutions by both methods for example 2 for different β^t values.

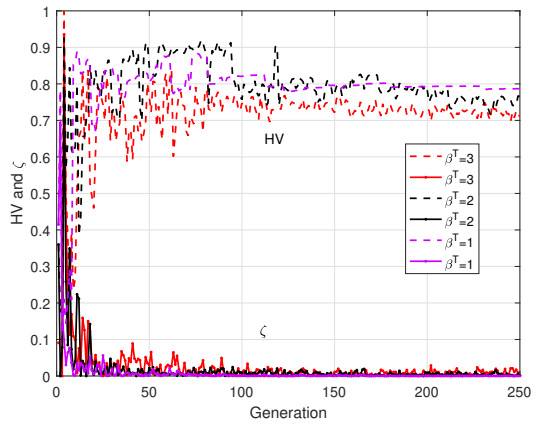


Figure 5.11: Progress of hypervolume and ζ of SLMDE with respect to number of generations for example 2

because PMA is performed for reliability estimation.

Figure 5.11 shows the progress of HV and ζ with respect to number of generations. It can be seen that there is fluctuations for all values of β^t till the termination criterion is achieved. Initial fluctuations can also be observed for ζ which subsidize after 150 generations.

5.6.3 Car side impact example

The car side impact example discussed in section 5.3.3 is considered as next engineering RBDO example, which is formulated by using two objectives and 10 performance functions. The

Table 5.13: Best, median, and worst HV values obtained by both methods are presented for car side impact example for different values of β^t. The best performances are highlighted in bold font.

β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE	β^t	SLMDE (Proposed)	DLMDE
	0.8251	0.8216		**0.7175**	0.7104		**0.5204**	0.5200
1.0	**0.8243**	0.8211	2.0	**0.7137**	0.7095	3.0	**0.5143**	0.5142
	0.8234	0.8196		**0.7078**	0.7070		**0.4957**	0.4912

Table 5.14: Number of function evaluations required by both methods for car side impact example.

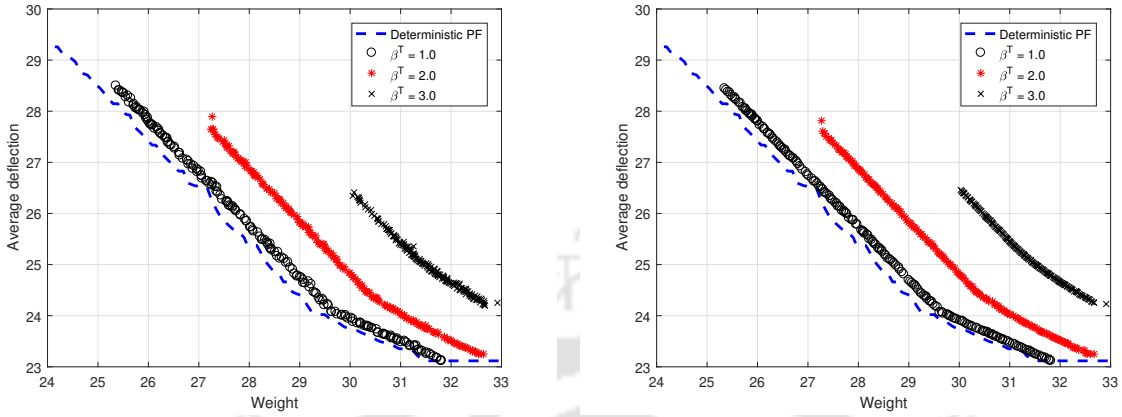
β^t	SLMDE (Proposed)	DLMDE
1.0	$46,23,000$	$1.467e+09$
2.0	$46,23,000$	$1.6970e+09$
3.0	$46,23,000$	$1.7342e+09$

RBDO formulation is presented in equation (5.11).

The statistical values of HV values obtained from both methods with respect to different β^t are presented in Table 5.13. The proposed method converges to larger values of HV for all β^t values. This signifies better distribution of PO solutions of SLMDE as compared to DLMDE. The observation of reducing HV values with larger β^t value remains the same.

The obtained reliable PO solutions for both methods are shown in Figure 5.12. As observed with previous examples, for larger values of β^t the PO solutions start moving away from the deterministic PO front inside the feasible region.

Table 5.14 presents the computational efficiency of both methods. The proposed method achieves a significant reduction in function evaluations compared to DLMDE, with an impressive range of approximately 317 to 377 times lesser evaluations required. Since, SLMDE performs approximate reliability estimation by using KKT optimality conditions, it saves many function evaluations compared to DLMDE. Figure 5.13 shows similar progress of HV and ζ with respect to number of generations. There are initial fluctuations for all values of β^t, which subside after 80 generations.



(a) SLMDE (b) DLMDE

Figure 5.12: The obtained PO solutions by both methods for car side impact example for different β^t values.

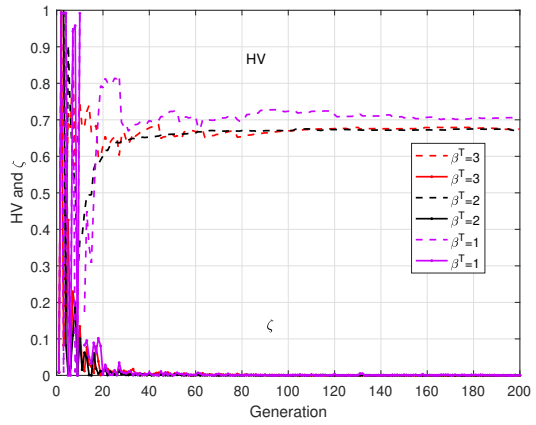


Figure 5.13: Progress of hypervolume and ζ of SLMDE with respect to number of generations for car side impact problem

5.7 Closure

In the last section, three RBDO examples were solved. All the examples consisted of two objective functions with nonlinear performance functions. The proposed SLMDE was able to generate reliable PO solutions with better computational efficiency compared to DLMDE. The solution spread of SLMDE was also better than DLMDE. The heuristic parameter ζ with generations also demonstrated stable progress for SLMDE. For example 2, a lot of fluctuations during the progress of HV can be observed which stabilises gradually. Comparing, the number

of function evaluations SLMDE demonstrated better performance than DLMDE. Both the SLMDE and DLMDE were tested with different values of β^t. It was found that SLMDE was able to generate better HV values compared to DLMDE for all the examples. It can also be observed that SLMDE shows smoother convergence than SL-MODE due to both the shifting vector approach and chaos control theory.

Chapter 6

Conclusion and future work

This chapter presents a general overview of the work done in this thesis. The aim of thesis was to develop efficient RBDO methods that can generate better reliable solutions as compared to the existing algorithms in the literature and can reduce the computational cost of these methods. Both numerical-based methods and meta-heuristic algorithms have been developed for solving single-objective and multi-objective RBDO examples. Uncertainties in the form of random variables were treated with low probability of failure of constraints to obtain reliable solution. All the developed methods were tested on several mathematical and engineering benchmark examples and compared with existing RBDO methods to validate the results.

6.1 Conclusions

The following are the important conclusion drawn from the thesis.

- As we have already discussed in the literature, the single-loop method was able to solve concave performance functions effectively and often faces the issue of convergence for concave and highly nonlinear performance functions. In chapter 3, two novel single-loop methods for solving single-objective RBDO examples using numerical optimization techniques were proposed which was able to solve both convex and concave performance functions effectively. The first method named SLShV-CG was developed for handing both convex and concave functions by approximating MPTP. The MPTP corresponding to each performance function was estimated with conjugate gradient search direction. This MPTP update strategy required the sensitivities from the previous two iterations. The shifting vector was coupled with SLShV-CG for generating computationally efficient

solutions. The method was tested with eight benchmark examples from the literature. From the results it can be concluded that SLShV-CG converged to the desired reliable solution irrespective of the nature of the performance function with better computational efficiency compared to the chosen set of RBDO methods. However, from example 2 it can be observed that SLSV-CG encounters with an issue of oscillation during convergence. As SLShV-CG method was developed using conjugate gradient search direction for approximating MPTP, it may have a tendency to oscillate while converging to the reliable solution for highly non-linear performance functions. This leads to the second objective of the thesis.

For addressing this issue ASLCC-2 was proposed in the second part of chapter 3. The oscillation criterion was proposed to estimate the chaos among consecutive MPTPs during convergence in the standard normal space. In every iteration, MPTP was estimated using conjugate gradient search directions and the last three MPTPs were used to track the oscillation among them. When oscillation was observed, chaos control theory was used to update the search direction of current MPTP. An oscillation criterion was also proposed to track these oscillations in the standard normal variable space. By solving the mathematical and engineering RBDO examples it can be observed that ASLCC-2 successfully eliminated the issue of oscillation while convergence. Example 2 also demonstrated smooth convergence when solved by ALSCC-2. Although, ALSCC-2 successfully mitigates the issue of oscillation, it has a tendency to converge to a local solutions. This leads to the motivation of third objective that is to obtain a reliable global solution.

- In the literature, there exists a single-loop method which was able to generate global solution. It was developed by using modified roulette wheel selection with stochastic acceptance in order to sort out solutions for mutation. Therefore, there was a need to explore the capabilities of other mutation schemes for obtaining a global reliable solution. In chapter 4, a single-loop reliability-based design optimization with adaptive DE has been proposed to address the issue of global convergence. A single-loop formulation was presented with shifting vector approach incorporating target and trial vectors. DE was made adaptive using the heuristic parameter that helped DE to perform different mutation operators. The proposed method, SLADE, was tested on three mathematical and four engineering RBDO examples. The results demonstrated that the proposed method successfully converged to the reliable global optimal solution for the chosen set of examples. The proposed method was also found computationally efficient than the

DLRBDO-DE method. Although, SLADE was able to generate reliable global solution, it was limited to single objective examples. It has been observed that most of the real-world problems consist of multiple objectives apart from uncertainty. This leads to the development of fourth objective that is an single-loop multi-objective reliability-based design optimization method.

- In the literature, many meta-heuristic algorithms were developed to obtain reliable PO solutions for multi-objective RBDO problems. All these methods were based on the platform of either double-loop or decoupled-loop method, which makes it computationally expensive. Therefore, efficient single-loop method should be developed to solve those problems. In chapter 5, for obtaining a set of reliable PO solutions a single-loop multi-objective formulation has been proposed incorporating chaos control theory for estimating the most probable target point. The formulation was solved using differential evolution with adaptive mutation scheme. A heuristic convergence parameter was proposed to perform different mutation schemes for better exploration and exploitation in the search space. The proposed SL-MODE was tested on two mathematical and one engineering bi-objective RBDO examples. It can be concluded from the results that the proposed SL-MODE generated the better reliable Pareto-optimal solutions than DL-MODE. Also, the convergence of SL-MODE observed less fluctuations than DL-MODE due to the estimation of MPTP using chaos control theory. Moreover, SL-MODE was found computationally efficient than DL-MODE by requiring less number of function evaluations. It has been observed in chapter 3 that both shifting vector approach and chaos control theory perform well solving RBDO examples. Therefore, in the last objective of this thesis both the concepts are incorporated for multi-objective RBDO formulation.

  In the last objective, a single-loop multi-objective reliability-based design optimization has been proposed with shifting vector approach for generating reliable PO solutions. The search direction of approximate MPTP was modified by chaos control theory. The concept of shifting vector approach was implemented with the novel multi-objective RBDO formulation to include both target and trial vectors. The proposed SLMDE was tested on the same mathematical and engineering bi-objective RBDO examples. It was found that SLMDE generated better reliable PO solutions for all examples compared to DLMDE.

From the above discussion, it can be concluded that this thesis is focused on development of single-loop reliability-based design optimization methods for solving single and multi-objective

optimization problem. Various issues related to single-loop method have been addressed in this study and the scope of future work is discussed in the next section.

6.2 Recommendation for future work

- In this thesis, the primary emphasis was placed on generating efficient and reliable solutions by incorporating KKT optimality conditions. However, it is important to acknowledge the limitations of this approach, such as its reliance on specific optimization techniques and assumptions. To address these limitations and further enhance the RBDO formulation, future research could explore alternative models like quantile approximation, surrogate modeling, dimension reduction methods, and other advanced techniques. Incorporating these approaches can potentially improve the efficiency and robustness of the single-loop RBDO formulation, opening new avenues for research in this field.

- In this thesis, we demonstrated the successful application of the developed methods to solve various mathematical and engineering RBDO examples. However, it is important to acknowledge the need for further validating the proposed methods on real-world problems. To enhance the robustness and applicability of our approach, future studies should incorporate a wider range of real-world engineering examples into the RBDO analysis. By doing so, we can gain deeper insights into the performance of our methods in practical scenarios and strengthen their relevance and effectiveness in industrial applications. Additionally, exploring the challenges and complexities posed by real-world engineering problems will provide valuable feedback for refining and improving our methods to better meet the needs of industry..

- In this thesis, the developed global RBDO method utilizing differential evolution demonstrates promising results. However, it is important to acknowledge the limitations of using a single evolutionary algorithm. Future studies should consider investigating alternative evolutionary algorithms to enhance the reliability of the optimal solutions and explore the strengths and weaknesses of different optimization techniques. By exploring a diverse range of evolutionary algorithms, researchers can better understand their capabilities and limitations in solving complex engineering problems. This exploration can lead to developing more effective and robust methods for RBDO, providing valuable insights and advancements in the field. Additionally, it is recommended to conduct comparative

studies that benchmark different evolutionary algorithms to identify their comparative performance and identify potential areas of improvement for future research.

References

Agarwal, H., C. K. Mozumder, J. E. Renaud, and L. T. Watson (2007). An inverse-measure-based unilevel architecture for reliability-based design optimization. *Structural and Multidisciplinary Optimization 33*(3), 217–227.

Aoues, Y. and A. Chateauneuf (2010, Mar). Benchmark study of numerical methods for reliability-based design optimization. *Structural and Multidisciplinary Optimization 41*(2), 277–294.

Biswas, R. and D. Sharma (2021a). An approximate single-loop chaos control method for reliability based design optimization using conjugate gradient search directions. *Engineering Optimization*, 1–17.

Biswas, R. and D. Sharma (2021b). A single-loop shifting vector method with conjugate gradient search for reliability-based design optimization. *Engineering Optimization 53*(6), 1044–1063.

Breitung, K. (1984, 03). Asymptotic approximations for multinormal integrals. *Journal of Engineering Mechanics 110*, 357–366.

Celorrio, L. and E. Patelli (2021). Reliability-based design optimization under mixed aleatory/epistemic uncertainties: Theory and applications. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering 7*(3), 04021026.

Chen, C.-T., M.-H. Chen, and W.-T. Horng (2014). A cell evolution method for reliability-based design optimization. *Applied Soft Computing 15*, 67–79.

Chen, J., T. Wang, J. Jia, L. Guo, and X. Wang (2022). Reliability analysis and abc based optimization for comp-enabled systems over nakagami-m fading. *Applied Soft Computing 117*, 108399.

Chen, X., T. Hasselman, D. Neill, X. Chen, T. Hasselman, and D. Neill (1997). Reliability based structural design optimization for practical applications. In *38th Structures, Structural Dynamics, and Materials Conference*, pp. 2724–2732.

Chen, Z., H. Qiu, L. Gao, and P. Li (2013a). An optimal shifting vector approach for efficient probabilistic design. *Structural and Multidisciplinary Optimization 47*(6), 905–920.

Chen, Z., H. Qiu, L. Gao, and P. Li (2013b). An optimal shifting vector approach for efficient probabilistic design. *Structural and Multidisciplinary Optimization 47*(6), 905–920.

Chen, Z., H. Qiu, L. Gao, X. Li, and P. Li (2014, 03). A local adaptive sampling method for reliability-based design optimization using kriging model. *Structural and Multidisciplinary Optimization 49*, 401–416.

Chen, Z., H. Qiu, L. Gao, L. Su, and P. Li (2013). An adaptive decoupling approach for reliability-based design optimization. *Computers and Structures 117*, 58 – 66.

Cheng, G., L. Xu, and L. Jiang (2006). A sequential approximate programming strategy for reliability-based structural optimization. *Computers & Structures 84*(21), 1353 – 1367.

Cho, T. M. and B. C. Lee (2011). Reliability-based design optimization using convex linearization and sequential optimization and reliability assessment method. *Structural Safety 33*(1), 42 – 50.

Choi, S.-K., R. Grandhi, and R. Canfield (2006, 01). *Reliability-based Structural Design*, Volume 1. Berlin, Germany: Springer.

Dammak, K. and A. E. Hami (2020). Multi-objective reliability based design optimization using kriging surrogate model for cementless hip prosthesis. *Computer Methods in Biomechanics and Biomedical Engineering 23*(12), 854–867. PMID: 32479190.

Deb, K. (2000). An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering 186*(2), 311–338.

Deb, K., S. Gupta, D. Daum, J. Branke, A. K. Mall, and D. Padmanabhan (2009). Reliability-based optimization using evolutionary algorithms. *IEEE Transactions on Evolutionary Computation 13*(5), 1054–1074.

Deb, K., A. Pratap, S. Agarwal, and T. Meyarivan (2002). A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation 6*(2), 182–197.

Du, X. and W. Chen (2004, 3). Sequential optimization and reliability assessment method for efficient probabilistic design. *Journal of Mechanical Design - Transactions of the ASME 126*(2), 225–233.

Elegbede, C. (2005). Structural reliability assessment based on particles swarm optimization. *Structural Safety 27*(2), 171–186.

Ezzati, G., M. Mammadov, and S. Kulkarni (2015, Jan). A new reliability analysis method based on the conjugate gradient direction. *Structural and Multidisciplinary Optimization 51*(1), 89–98.

Forouzandeh Shahraki, A. and R. Noorossana (2014). Reliability-based robust design optimization: A general methodology using genetic algorithm. *Computers & Industrial Engineering 74*, 199–207.

Gargama, H., S. K. Chaturvedi, and A. K. Thakur (2014). Reliability-based design optimization of electromagnetic shielding structure using neural networks and real-coded genetic algorithm. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science 228*(18), 3471–3481.

Goldberg, D. (1989). *Genetic Algorithms in Search Optimization and Machine Learning*. 75 Arlington Street, Suite 300 Boston, MA, United States: Addison-Wesley Longman Publishing Co., Inc.

Grandhi, R. V. and L. Wang (1998). Reliability-based structural optimization using improved two-point adaptive nonlinear approximations. *Finite Elements in Analysis and Design 29*(1), 35–48.

Hamzehkolaei, N. S., M. Miri, and M. Rashki (2018). New simulation-based frameworks for multi-objective reliability-based design optimization of structures. *Applied Mathematical Modelling 62*, 1–20.

Hasofer, A. M. and N. Lind (1974). Exact and invariant second-moment code format. *Journal of Engineering Mechanics-ASCE 100*, 111–121.

Ho-Huu, V., T. Le-Duc, L. Le-Anh, T. Vo-Duy, and T. Nguyen-Thoi (2018). A global single-loop deterministic approach for reliability-based design optimization of truss structures with continuous and discrete design variables. *Engineering Optimization 50*(12), 2071–2090.

Ho-Huu, V., T. Truong, L. Le, and T. Vo-Duy (2018, 01). An improved differential evolution based on roulette wheel selection for shape and size optimization of truss structures with frequency constraints. *Neural Computing and Applications 29*, 167185.

Hock, W. and K. Schittkowski (1981). *Test Examples for Nonlinear Programming Codes*. Berlin, Heidelberg: Springer-Verlag.

Huang, H.-Z., X. Zhang, D.-B. Meng, Z. Wang, and Y. Liu (2013). An efficient approach to reliability-based design optimization within the enhanced sequential optimization and reliability assessment framework. *Journal of Mechanical Science and Technology 27*(6), 1781–1789.

Huang, Z. L., C. Jiang, Y. S. Zhou, Z. Luo, and Z. Zhang (2016). An incremental shifting vector approach for reliability-based design optimization. *Structural and Multidisciplinary Optimization 53*(3), 523–543.

Jafari-Asl, J., M. E. A. Ben Seghier, S. Ohadi, and P. van Gelder (2021). Efficient method using whale optimization algorithm for reliability-based design optimization of labyrinth spillway. *Applied Soft Computing 101*, 107036.

Jeong, S.-B. and G.-J. Park (2017). Single loop single vector approach using the conjugate gradient in reliability based design optimization. *Structural and Multidisciplinary Optimization 55*(4), 1329–1344.

Jiang, C., H. Qiu, L. Gao, X. Cai, and P. Li (2017a, December). An adaptive hybrid single-loop method for reliability-based design optimization using iterative control strategy. *Structural and Multidisciplinary Optimization 56*(6), 1271–1286.

Jiang, C., H. Qiu, L. Gao, X. Cai, and P. Li (2017b). An adaptive hybrid single-loop method for reliability-based design optimization using iterative control strategy. *Structural and Multidisciplinary Optimization 56*(6), 1271–1286.

Karaboga, D. and B. Basturk (2007). Artificial bee colony (abc) optimization algorithm for solving constrained optimization problems. In P. Melin, O. Castillo, L. T. Aguilar, J. Kacprzyk,

and W. Pedrycz (Eds.), *Foundations of Fuzzy Logic and Soft Computing*, Berlin, Heidelberg, pp. 789–798. Springer Berlin Heidelberg.

Kennedy, J. and R. Eberhart (1995). Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, Volume 4, pp. 1942–1948.

Keshtegar, B. and P. Hao (2016, 09). A Hybrid Loop Approach Using the Sufficient Descent Condition for Accurate, Robust, and Efficient Reliability-Based Design Optimization. *Journal of Mechanical Design 138*(12), 1 – 11.

Keshtegar, B. and P. Hao (2017). A hybrid self-adjusted mean value method for reliability-based design optimization using sufficient descent condition. *Applied Mathematical Modelling 41*, 257 – 270.

Keshtegar, B., P. Hao, and Z. Meng (2017, January). A self-adaptive modified chaos control method for reliability-based design optimization. *Structural and Multidisciplinary Optimization 55*(1), 63–75.

Keshtegar, B. and I. Lee (2016). Relaxed performance measure approach for reliability-based design optimization. *Structural and Multidisciplinary Optimization 54*(6), 1439–1454.

Khodam, A., P. Mesbahi, M. Shayanfar, and B. M. Ayyub (2021). Global decoupling for structural reliability-based optimal design using improved differential evolution and chaos control. *ASCE-ASME Journal of Risk and Uncertainty in Engineering Systems, Part A: Civil Engineering 7*(1), 04020052.

Kiureghian, A. D., H. Lin, and S. Hwang (1987). Second order reliability approximations. *Journal of Engineering Mechanics 113*(8), 1208–1225.

Kogiso, N., Y. Young-Soon, K. Bong-Jae, and L. Jae-Ohk (2012). Modified single-loop-single-vector method for efficient reliability-based design optimization. *Journal of Advanced Mechanical Design, Systems, and Manufacturing 6*(7), 1206–1221.

Kuschel, N. and R. Rackwitz (1997, October). Two basic problems in reliability-based structural optimization. *Mathematical Methods of Operations Research 46*(3), 309–333.

Lee, J. J. and B. C. Lee (2005). Efficient evaluation of probabilistic constraints using an envelope function. *Engineering Optimization 37*(2), 185–200.

Li, F., G. Sun, X. Huang, J. Rong, and Q. Li (2015). Multiobjective robust optimization for crashworthiness design of foam filled thin-walled structures with random and interval uncertainties. *Engineering Structures 88*, 111–124.

Li, F., T. Wu, A. Badiru, M. Hu, and S. Soni (2013). A single-loop deterministic method for reliability-based design optimization. *Engineering Optimization 45*(4), 435–458.

Li, F., T. Wu, M. Hu, and J. Dong (2010). An accurate penalty-based approach for reliability-based design optimization. *Research in Engineering Design 21*(2), 87–98.

Li, G., Z. Meng, and H. Hu (2015, 05). An adaptive hybrid approach for reliability-based design optimization. *Structural and Multidisciplinary Optimization 51*, 1051–1065.

Li, X., C. Gong, L. Gu, Z. Jing, H. Fang, and R. Gao (2019, 02). A reliability-based optimization method using sequential surrogate model and monte carlo simulation. *Structural and Multidisciplinary Optimization 59*, 439–460.

Liang, J., Z. Mourelatos, and J. Tu (2008). A single-loop method for reliability-based design optimization. *International Journal of Product Development 5*(1-2), 76–92.

Liao, K. W. and G. Ivan (2014, 10). A single loop reliability-based design optimization using epm and mpp-based pso. *Latin American Journal of Solids and Structures 11*, 826–847.

Lim, J. and B. Lee (2016). A semi-single-loop method using approximation of most probable point for reliability-based design optimization. *Structural and Multidisciplinary Optimization 53*(4), 745–757.

Lim, J., B. Lee, and I. Lee (2014). Second-order reliability method-based inverse reliability analysis using hessian update for accurate and efficient reliability-based design optimization. *International Journal for Numerical Methods in Engineering 100*(10), 773–792.

Lim, J., B. Lee, and I. Lee (2016). Post optimization for accurate and efficient reliability-based design optimization using second-order reliability method based on importance sampling and its stochastic sensitivity analysis. *International Journal for Numerical Methods in Engineering 107*(2), 93–108.

Liu, P.-L. and A. D. Kiureghian (1986). Multivariate distribution models with prescribed marginals and covariances. *Probabilistic Engineering Mechanics 1*(2), 105 – 112.

Liu, S., X. Wang, J. Kong, J. Zhang, and W. Tang (2023, 04). Kriging-based performance measure function approximation method for hybrid reliability-based design optimization. *IEEE Access*, 1–1.

Lobato, F. S., M. S. Gonalves, B. Jahn, A. A. Cavalini, and V. Steffen (2017). Reliability-based optimization using differential evolution and inverse reliability analysis for engineering system design. *Journal of Optimization Theory and Applications 174*(3), 894–926.

Madsen, H., S. Krenk, and N. Lind (1986). *Methods of Structural Safety*. Mineola, New York: Dover Publication, Inc.

Meng, Z. and B. Keshtegar (2018, 10). Adaptive conjugate single-loop method for efficient reliability-based design and topology optimization. *Computer Methods in Applied Mechanics and Engineering 344*, 95–119.

Meng, Z., G. Li, B. P. Wang, and P. Hao (2015). A hybrid chaos control approach of the performance measure functions for reliability-based design optimization. *Computers & Structures 146*, 32 – 43.

Meng, Z., A. Rza Yldz, and S. Mirjalili (2022). Efficient decoupling-assisted evolutionary/metaheuristic framework for expensive reliability-based design optimization problems. *Expert Systems with Applications 205*, 117640.

Meng, Z., D. Yang, H. Zhou, and B. Yu (2018). An accurate and efficient reliability-based design optimization using the second order reliability method and improved stability transformation method. *Engineering Optimization 50*(5), 749–765.

Meng, Z., H. Zhou, G. Li, and H. Hu (2017). A hybrid sequential approximate programming method for second-order reliability-based design optimization approach. *Acta Mechanica 228*(5), 1965–1978.

Muhuri, P., Z. Ashraf, and Q. M. D. Lohani (2018, 06). Multi-objective reliability-redundancy allocation problem with interval type-2 fuzzy uncertainty. *IEEE Transactions on Fuzzy Systems 26*, 1339–1355.

Nikolaidis, E. and R. Burdisso (1988). Reliability based optimization: A safety index approach. *Computers & Structures 28*(6), 781 – 788.

Osyczka, A. and S. Kundu (1995). A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm. *Structural optimization 10*, 94–99.

Periaro, G., S. Santos, A. Ribeiro, and L. Matioli (2015). Hlrfbfgs optimization algorithm for structural reliability. *Applied Mathematical Modelling 39*(7), 2025–2035.

Pingel, D., P. Schmelcher, and F. Diakonos (2004, 10). Stability transformation: A tool to solve nonlinear problems. *Physics Reports-review Section of Physics Letters 400*, 67–148.

Putko, M. M., I. Taylor, Arthur C., P. A. Newman, and L. L. Green (2001, 11). Approach for Input Uncertainty Propagation and Robust Design in CFD Using Sensitivity Derivatives1. *Journal of Fluids Engineering 124*(1), 60–69.

Qu, X. and R. Haftka (2004, 07). Reliability-based design optimization using probabilistic sufficiency factor. *Structural and Multidisciplinary Optimization 5*(27), 314–325.

Rackwitz, R. and B. Flessler (1978). Structural reliability under combined random load sequences. *Computers & Structures 9*(5), 489 – 494.

Ramu, P., X. Qu, B. D. Youn, R. Haftka, and K. Choi (2006, 01). Inverse reliability measures and reliability-based design optimisation. *International Journal of Reliability and Safety 1*(1/2), 187–205.

Rao, S. (2009, 6). *Engineering Optimization: Theory and Practice: Fourth Edition*. John Wiley and Sons.

Reddy, M., R. Grandhi, and D. Hopkins (1994). Reliability based structural optimization: A simplified safety index approach. *Computers & Structures 53*(6), 1407 – 1418.

Rosenblatt, M. (1952). Remarks on a Multivariate Transformation. *The Annals of Mathematical Statistics 23*(3), 470 – 472.

Safaeian Hamzehkolaei, N., M. Miri, and M. Rashki (2016). An enhanced simulation-based design method coupled with meta-heuristic search algorithm for accurate reliability-based design optimization. *Engineering with Computers 32*(3), 477–495.

Shan, S. and G. G. Wang (2008). Reliable design space and complete single-loop reliability-based design optimization. *Reliability Engineering & System Safety 93*(8), 1218–1230.

Shayanfar, M., R. Abbasnia, and A. Khodam (2014). Development of a ga-based method for reliability-based optimization of structures with discrete and continuous design variables using opensees and tcl. *Finite Elements in Analysis and Design 90*, 61–73.

Shi, Z.-J., S. Wang, and Z. Xu (2010). The convergence of conjugate gradient method with nonmonotone line search. *Applied Mathematics and Computation 217*(5), 1921 – 1932.

Shirgir, S., A. Shamsaddinlou, R. N. Zare, S. Zehtabiyan, and M. H. Bonab (2023). An efficient double-loop reliability-based optimization with metaheuristic algorithms to design soil nail walls under uncertain condition. *Reliability Engineering & System Safety 232*, 109077.

Sleesongsom, S. and S. Bureerat (2020). Multi-objective, reliability-based design optimization of a steering linkage. *Applied Sciences 10*(17).

Storn, R. and K. Price (1997). Differential evolution a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization 11*(4), 341–359.

Sun, G., H. Zhang, J. Fang, G. Li, and Q. Li (2017, May). Multi-objective and multi-case reliability-based design optimization for tailor rolled blank (trb) structures. *Structural and Multidisciplinary Optimization 55*(5), 1899–1916.

Sun, G., H. Zhang, R. Wang, X. Lv, and Q. Li (2017, 12). Multiobjective reliability-based optimization for crashworthy structures coupled with metal forming process. *Structural and Multidisciplinary Optimization 56*, 1571–1587.

Torii, A. J., R. H. Lopez, and L. F. Miguel (2016). A general rbdo decoupling approach for different reliability analysis methods. *Structural and Multidisciplinary Optimization 54*(2), 317–332.

Truong, V.-H. and S.-E. Kim (2018). Reliability-based design optimization of nonlinear inelastic trusses using improved differential evolution algorithm. *Advances in Engineering Software 121*, 59–74.

Tu, J., K. K. Choi, and Y. H. Park (1999, 12). A New Study on Reliability-Based Design Optimization. *Journal of Mechanical Design 121*(4), 557–564.

Tvedt, L. (1983). Two second-order approximations to the failure probability-section on structural reliability. *Technical Report No. RDIV/20-004-83. A/S Veritas Research, Hovik, 1984*.

Weiji, L. and Y. Li (1994). An effective optimization procedure based on structural reliability. *Computers & Structures 52*(5), 1061–1067.

Wu, Y.-T., H. R. Millwater, and T. A. Cruse (1990). Advanced probabilistic structural analysis method for implicit performance functions. *American Institute of Aeronautics and Astronautics 28*, 1663 – 1669.

Yang, D. and P. Yi (2009). Chaos control of performance measure approach for evaluation of probabilistic constraints. *Structural and Multidisciplinary Optimization 38*(1), 83–92.

Yang, I.-T. and Y.-H. Hsieh (2013, 04). Reliability-based design optimization with cooperation between support vector machine and particle swarm optimization. *Engineering with Computers 29*, 151–163.

Yi, P., Z. Zhu, and J. Gong (2016). An approximate sequential optimization and reliability assessment method for reliability-based design optimization. *Structural and Multidisciplinary Optimization 54*(6), 1367–1378.

Youn, B. D., K. Choi, and Y. Park (2003). Hybrid analysis method for reliability-based design optimization. *Journal of Mechanical Design 125*(2), 221–232.

Yu, S., Z. Wang, and Z. Wang (2019, 04). Time-Dependent Reliability-Based Robust Design Optimization Using Evolutionary Algorithm. *ASCE-ASME J Risk and Uncert in Engrg Sys Part B Mech Engrg 5*(2). 020911.

Zafar, T., Y. Zhang, and Z. Wang (2020). An efficient kriging based method for time-dependent reliability based robust design optimization via evolutionary algorithm. *Computer Methods in Applied Mechanics and Engineering 372*, 113386.

Zhang, H., Y. Aoues, H. Bai, D. Lemosse, and E. S. de Cursi (2020). Kriging-based reliability-based design optimization using single loop approach. In *Optimization of Complex Systems: Theory, Models, Algorithms and Applications*, Cham, pp. 991–1000. Springer International Publishing.

Zhao, H., M. Zhao, and C. Zhu (2016). Reliability-based optimization of geotechnical engineering using the artificial bee colony algorithm. *KSCE Journal of Civil Engineering 20*, 1728–1736.

Zhu, Z.-Z., Y.-W. Feng, C. Lu, and C.-W. Fei (2020). Reliability optimization of structural deformation with improved support vector regression model. *Advances in Materials Science and Engineering Article ID 3982450*, 1687–8434.