2016

Genetic Algorithms in Stochastic Optimization and Applications in Power Electronics

Mengmei Liu
University of Kentucky, mengmei.liu@gmail.com

Digital Object Identifier: https://doi.org/10.13023/ETD.2016.475

Genetic Algorithms in Stochastic Optimization and Applications in Power electronics

DISSERTATION

A dissertation submitted in partial fulfillment of the requirement for the degree of Doctor of Philosophy in the College of Engineering at the University of Kentucky

By

Mengmei Liu

Lexington, Kentucky

Director: Dr. Aaron M. Cramer, Associate Professor of Electrical Engineering

Lexington, Kentucky

2016

ABSTRACT OF DISSERTATION

Genetic Algorithms in Stochastic Optimization and Applications in Power electronics

Genetic Algorithms (GAs) are widely used in multiple fields, ranging from mathematics, physics, to engineering fields, computational science, bioinformatics, manufacturing, economics, etc. The stochastic optimization problems are important in power electronics and control systems, and most designs require choosing optimum parameters to ensure maximum control effect or minimum noise impact; however, they are difficult to solve using the exhaustive searching method, especially when the search domain conveys a large area or is infinite. Instead, GAs can be applied to solve those problems. And efficient computing budget allocation technique for allocating the samples in GAs is necessary because the real-life problems with noise are often difficult to evaluate and require significant computation effort. A single objective GA is proposed in which computing budget allocation techniques are integrated directly into the selection operator rather than being used during fitness evaluation. This allows fitness evaluations to be allocated towards specific individuals for whom the algorithm requires more information, and this selection-integrated method is shown to be more accurate for the same computing budget than the existing evaluation-integrated methods on several test problems. A combination of studies is performed on a multi-objective GA that compares integration of different computing budget allocation methods into either the evaluation or the environmental selection steps. These comparisons are performed on stochastic problems derived from benchmark multi-objective optimization problems and consider varying levels of noise. The algorithms are compared regarding both proximity to and coverage of the true Pareto-optimal front, and sufficient studies are performed to allow statistically significant conclusions to be drawn. Finally, the multi-objective GA with selection integrated sampling technique is applied to solve a multi-objective stochastic optimization problem in a grid connected photovoltaic inverter system with noise injected from both the solar power input and the utility grid.

KEYWORDS: genetic algorithms, stochastic optimization, photovoltaic systems, computing budget allocation, PR control, virtual resistance damping.

Mengmei Liu

Student's Signature

December 9th, 2016

Date

Genetic Algorithms in Stochastic Optimization and Applications in Power electronics

By

Mengmei Liu

Aaron M. Cramer

Director of Dissertation

Caicheng Lu

Director of Graduate Studies

December 9th, 2016

This dissertation is gratefully dedicated to my mother, Honglan Wang.

ACKNOWLEDGEMENTS

First, I would like to thank my advisor Prof. Aaron Cramer for his guidance through the dissertation research. I am really grateful for working on these interesting topics under his support and continuous encouragement. This work would not be possible without his outstanding knowledge and innovative ideas.

Second, I would like to thank the rest of the committee members: Prof. Bruce Walcott, Prof. Sen-ching Samson Cheung and Prof. Zongming Fei for their thorough comments and insightful suggestions. I would also like to thank Dr. Benjamin Loop, for his help during the dissertation study.

Third, I would like to thank my fellow colleagues, Xiao Liu, Fei Pan and Hanling Chen, for their generous help during the experiments.

Last but not the least, I would like to thank my parents and grandparents for their love and support.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Genetic Algorithms (GAs) are widely used in multiple fields, ranging from mathematics, physics, to engineering fields, computational science, manufacturing, even to economics [1], etc. The stochastic optimization problems are important in power electronics and control systems [2], and most designs require choosing optimum parameters to ensure maximum control effect or minimum noise impact; however, they are tough to solve using the exhaustive searching method, especially when the search domain conveys a large area or is infinite [3]. Earlier researchers had proposed random search algorithms, such as the Stochastic Ruler (SR) algorithm [4], and the Stochastic Comparison (SC) algorithm [5]. However, these algorithms are difficult for application problems because it is hard to decide the termination of the algorithms. Recently, meta-heuristics methods, such as Genetic Algorithms, Particle Swarm Algorithms, and Simulated Annealing, have drawn attention to researchers. A combination of the searching method with statistical analysis techniques has made progress in solving noisy function optimization.

Christian Schmidtl et al. presented integrating techniques from Statistical Ranking into Evolutionary Algorithms [6], Hui Pan et al. presented a combination of Particle Swarm Optimization with Optimal Computing Budget Allocation (OCBA) technique [7], Alkhamis and Ahmed performed simulated annealing with confidence intervals for simulation-based optimization [8].

The genetic algorithm is the most widely used meta-heuristics method to solve stochastic optimization problems [9] [10]. Starting from a random search with no prior information, the genetic operators guide the evolution of chromosomes (which represent the solution set in this case) to the optimum solution. Sampling allocation schemes are required to estimate the noise when evaluating stochastic problems; however, in the ranking and selection process, one does not want to waste samples for the solution that is unlikely to be the optimum solution or to oversample when the noise level is low. In this case, computer budget allocation schemes are introduced and integrated into the genetic algorithms to solve stochastic optimization problems.

Traditionally, the genetic algorithm evaluates all the individuals once at the evaluation process. Then the selection process is only used for selecting the optimum off-spring from the fitness value computed in the evaluation. However, the selection procedure is also a key factor where ranking occurs. Thus, in stochastic problems, the probability of correct selection in the selection procedure can be improved by allocating samples during selection. And it may lead to improvement of overall performance in the genetic algorithm.

There are two different ways of introducing the computer budget allocation schemes into genetic algorithms with same initial sampling guaranteed in both cases. The first is to allocate the computing budget into the evaluation procedure [11] [6] to further improve the accuracy of the evaluation. The second is to allocate the computing budget into the selection procedure to further improve the accuracy of selection.

In this dissertation, a comprehensive study of the different computing budget allocation schemes is included for both single objective and multi-objective genetic algorithms. Details of the integration will be introduced in later chapters. In both cases, typical test problems are studied in different noise level, and statistically validated results are produced from large amount experiments and statistical tests. It is found that while different computing budget schemes can affect the results in various test problems, it is more general that selection integrated sampling technique performs better in genetic algorithm than evaluation integrated sampling technique in both single objective genetic algorithm and multi-objective genetic algorithm.

Based on this conclusion, selection integrated sampling genetic algorithm can be applied to solve practical power electronics and control problems. Many power electronics problems are discrete event stochastic optimization problems with unknown noise. The only way to evaluate the system performance is to perform independent experiments or simulations. However, for complicated control systems, it is computationally intensive to run those simulations. In this case, an algorithm which provides the optimum solution for the single/multi-objective stochastic optimization problems with the least number of simulation/evaluation is necessary. The selection integrated genetic algorithm is applied to solve such control problems and provides the optimum design solution for power electronics devices.

1.1 Applications of Dissertation

The first application of this dissertation is to improve the efficiency of a genetic algorithm on single-objective stochastic problems by integrating the computing budget allocation technique into the tournament selection process. Different allocation techniques will be utilized and compared. Multiple deterministic test problems with a known global optimum point are being added with a zero mean Gaussian noise to test with the integrated genetic algorithm.

The second application of this dissertation is to improve the efficiency of a multi-objective genetic algorithm to solve multi-object stochastic problems by integrating the computing budget allocation technique into the environmental selection process. For a multi-objective problem, there is no fixed optimum solution. However, we can find the Pareto optimum solution, which is the set of solutions that cannot improve any objective without sacrificing other objectives. Also, using a genetic algorithm with computing allocation budget integrated into the environmental selection process will improve the efficiency of finding Pareto optimum solutions for multi-objective stochastic problems.

The third application focuses on applying the selection integrated genetic algorithm to solve a realistic stochastic power electronic problem. Optimization of grid-connected photovoltaic inverter system is considered a complicated stochastic problem due to the randomness in solar irradiance and the grid variations. The system is very complex and takes a long time for one simulation. It is important to locate optimum control parameters to maintain the system stability and performance in varies noise environment. Thus a significant amount of simulation in different noise cases are required to select optimum results. And the proposed algorithm can be applied to improve the accuracy of results given that computing power is limited. Hardware experiment is performed to validate the algorithm results.

1.2 Organization

In this dissertation, the integrated genetic algorithms are proposed to solve stochastic optimization problems, and a stochastic multi-objective power electronics problem is solved through the proposed algorithm. The remainder of this dissertation is organized as follows:

Chapter 2 gives the background and literature review of the single objective genetic algorithm, multi-objective genetic algorithm, optimum computing budget allocation schemes and photovoltaic inverter.

Chapter 3 illustrates different methods to integrate computing budget allocation schemes into the single objective genetic algorithm, which are the selection integrated method and the evaluation integrated method. The basic operators of a single objective genetic algorithm include initialization, evaluation, tournament selection, crossover, mutation, elitism, and termination. The selection integrated genetic algorithm integrates the computing budget allocation into the tournament selection procedure, while the evaluation integrated method integrates the computing budget allocation into the evaluation process. Experiments on test problems are conducted, and the results show that statistically the selection integrated genetic algorithm performed than the evaluation integrated method.

Chapter 4 illustrates different methods to integrate computing budget allocation schemes into the multi objective genetic algorithm, which are the selection integrated method and the evaluation integrated method. The basic procedures of a multi-objective genetic algorithm are different from a single objective algorithm which include two different selection procedures. The basic workflow of a multi-objective genetic algorithm includes initialization which includes initial evaluation to produce pool, mating selection, which is identical to tournament selection in the single objective genetic algorithm, followed by crossover, mutation, evaluation to produce a new pool. Then the new pool is combined with old pool for environmental selection, which ranks and selects the next generation pool. The environmental selection is a more major selection procedure than mating selection in the multi-objective genetic algorithm, so the selection integrated method for the multi-objective genetic algorithm is integrated into the environmental selection. The evaluation integrated method remains the same as the single objective genetic algorithm. Experiments on multi-objective test problems are also performed and similar conclusion of a better result from selection integrated method can also be drawn statistically.

Chapter 5 illustrates a practical multi-objective stochastic power electronic problem and solve the problem with the proposed selection integrated multi-objective genetic algorithm. Randomness from solar irradiance and grid variations are introduced to a photovoltaic inverter system. The randomness from solar irradiance is introduced in both input power and ramp power to mimic the current solar irradiance and irradiance change per second. The randomness from grid variation is introduced in both base grid voltage magnitude and 5th and 7th harmonics voltage magnitude. The two optimization objects are to minimize accumulative voltage variation from simulation which includes a one-second power ramp and to minimize the grid current. Optimization goals are to be achieved through tuning of control parameters using the multi-objective genetic algorithm. The optimal results are validated and tested with hardware experiments.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

Based on the previous introduction, genetic algorithms, different computing budget allocation methods, and realistic power electronic problems are the three key factors in this dissertation. Related background information and literature review will be presented. The remainder of this chapter is organized as follows: background of the single-objective genetic algorithm is introduced in section 2.1; followed by the background of the multi-objective genetic algorithm in section 2.2. Optimal computing budget techniques [12] with supportive lemmas are introduced in section 2.3. Buck converter is introduced in section 2.4. The photovoltaic inverter is introduced in section 2.5.

2.1 Single-objective Genetic Algorithm

Genetic Algorithms (GAs) are searching and optimization algorithms that mimic the process of natural selection described by Charles Darwin. The concept of GAs was first introduced by John Holland from University of Michigan, Ann Arbor, and it was developed rapidly afterward [13]. GAs are naturally parallel and direct methods which aim to evaluate and modify sets of solutions simultaneously [14]. After initialization, a single-objective GA, which will be referred as SOGA in this section, repeats the basic procedures of evaluation, selection, crossover, mutation, and elitism until termination criteria are met. Detailed information about each procedure is given as follows.

2.1.1 Chromosome encoding

In natural selection, a chromosome represents the gene with the encoded information that determines distinct properties. Similarly, chromosomes encode the independent decision variables (also known as genes) in genetic algorithms [14]. Each gene represents a unique design in the process. For the optimization problem, the coding scheme is:

Chromosome: $\hat{X} : X_1, X_2, \ldots, X_k$, where k is the number of designs, and $X_i (i = 1, 2, \ldots, k)$ is the solution vector that represents the ith gene.

2.1.2 Initialization

The SOGA starts with initialization. During this process, random values in the search space are assigned to each chromosome. Specifically, the initial random chromosomes are uniformly distributed between the maximum and minimum search intervals to form the initial pool which is evaluated with the number of initial samples.

2.1.3 Selection

Different selection algorithms are adapted to different SOGAs, and tournament selection is one of the most popular methods [15] [16]. The tournament selection can be processed in three steps. First, randomly select m individuals from the pool, where m is the tournament size and must be greater than or equal to 2. Second, rank the fitness value for the tournament individuals. Third, choose the individual with the highest rank to generate the new pool. For example, if the pool size is n, then n different tournaments will be generated randomly, with the highest rank generating a new pool for the next generation. The tournament size m determines the selection pressure: the larger m is, the more difficult it is to select an individual with weak fitness value.

2.1.4 Crossover

Crossover is the first step of reproduction; it uses the genes of the parents to form new chromosomes. In this study, simulated binary crossover [17] [18] is used.

Simulated binary crossover (SBX) is similar to one-point crossover in binary SOGAs; however, it performs as well or better than the one-point method. SBX is particularly useful in solving complicated problems, such as problems with multi-optimum solutions.

SBX generates two children, c_1 and c_2, from two parent genes, p_1 and p_2, using a random generated spread factor, β, according to the following:

$$c_1 = \bar{x} - \frac{1}{2}\bar{\beta}(p_2 - p_1) \tag{2.1}$$

$$c_1 = \bar{x} + \frac{1}{2}\bar{\beta}(p_2 - p_1) \tag{2.2}$$

And the proposed probability distribution function of β is:

$$c(\beta) = 0.5(n+1)\beta^n, \beta \leq 1 \tag{2.3}$$

$$c(\beta) = 0.5(n+1)\frac{1}{\beta^{n+2}}, \beta > 1 \tag{2.4}$$

2.1.5 Mutation

In SOGAs, the mutation may randomly alter the value of one or more genes from the population. The changes mimic the natural process by which mistakes are copied from the parent genes to the children.

The process of mutation can follow the process of crossover. For example, a gene to be mutated is first randomly selected, and the simulated binary crossover is used to correlate the gene with a randomly generated number to ensure the randomness of mutation.

2.1.6 Evaluation

The evaluation process is necessary to assign a fitness value to each chromosome, thereby ranking gene performance. The fitness value guides the evolution of a chromosome from generation to generation. Typically, evaluation calculates the fitness function, sometimes through running a simulation for a discrete system. For the stochastic problems considered in this dissertation, the noise vectors are also introduced in the evaluation process.

2.1.7 Insertion

After the process of tournament selection, crossover and mutation, the new mating pool was formed. The insertion process randomly insert the new mating pool into previous pool to generate the next generation pool.

2.1.8 Termination

The SOGA terminates if the maximum iteration occurs. During the SOGA process, the information of chromosomes and fitness value from each generation is collected. Those data are to be analyzed after termination.

2.2 Multi-objective Genetic Algorithm

In the last section, background and basic procedures of the single-objective genetic algorithm are introduced. However, in real-world optimization, application problems often contain more than one objectives that need to be optimized. For example, an economic power dispatch problem which aims to minimize and emission under constraints of overall capacity, power balance and security constraints is described in [19]. A multi-objective optimization issue, which demands minimal of initial cost or the maximum of power density and the maximum of power conversion efficiency with a possible energy cost of a single phase rectifier is proposed in [20]. A multi-objective control goal for the power converter in active hybrid fuel cell/battery sources of energy includes regulating the output current of the fuel cell and the charging current of battery while limiting the discharging current of the battery is achieved in [21] with a flexible multi-objective control strategy. In [22], multi-objective genetic algorithms (MOGAs) is demonstrated to be one of the efficient approaches to solving such multi-objective engineering problems.

MOGAs usually combine operators such as mating selection [23], which selects child genes, crossover, and mutation to construct new generations of individuals. Among MOGAs, most elitist approaches archive non-dominated solutions from the previous generation and combine them with non-dominated solutions from the current generation to produce the subsequent generation, a process which is referred to as environmental selection [23]. A non-dominated sorting based MOGA, which is called NSGA-II (Non-dominated Sorting Genetic Algorithm II) is one of the most widely used and complexity efficient methods among all the MOGAs [18]. The general procedure of an NSGA-II will be introduced herein.

2.2.1 Chromosome encoding

For MOGA, the search domain is $\mathcal{R}^n$ where n is the number of objectives. For an objective i, the chromosomes are represented by $\hat{X}_i : X_{i1}, X_{i2}, \ldots, X_{ik}$, where k is the number of designs, and $X_{ij}(i = 1, 2, \ldots, n,\ j = 1, 2, \ldots, k)$ is the solution vector that represents the j-th gene of i-th objective.

2.2.2 Initialization

The MOGA also starts with random initialization similar to SOGA. Unlike to SOGA, the search domain is multi-dimensional. For two-objective optimization problems, the search domain forms a plane; for three objective problems, the search domain forms a 3D space and can be imagined as unbounded cubic; For higher dimension, the search domain forms a higher dimensional space and can hardly be imagined. The initial random chromosomes are uniformly distributed between the maximum and minimum search intervals in each dimension and build a n dimensional chromosomes space.

2.2.3 Evaluation and Non-dominated Sorting

In MOGA, evaluation assigns fitness value to each objective. For SOGA, the ranking and comparison of two individuals are determined by its fitness. However, for MOGA, it is difficult to determine which individual is best by merely observing the fitness. In the multi-objective scenario, correctly ranking the individuals based on the fitness value and assigning a legitimate rank to discriminate the individual becomes an important issue. A fast non-dominated sorting algorithm was first proposed by Deb.et al [18] in NSGA-II as an enhancement to NSGA and was widely used for its accuracy and efficiency in determining the rank. The sorting algorithm is described as follows:

Crowding distance is another measurement for an individual calculated by its fitness values. If two individuals have the same rank, crowding distance is used to discriminate the two individuals. The basic idea for calculating the crowding distance is finding the Euclidean distance between each in a front based on the multi-dimensional hyperspace. The individuals in the boundary are assigned with infinite distance. The algorithm for calculating the crowding distance is described as follows:

Algorithm 1 Non-dominated sort

```
1: for Each individual $p$ in main polulation $P$ do
2:     Initialize $S_p = \emptyset$. $S_p$ denotes the set that is dominated by $p$.
3:     Initialize $n_p = 0$. $n_p$ denotes the number of individuals that dominate $p$.
4:     for Each individual $q \in P$ do
5:         if $p$ dominates $q$ then
6:             Add $q$ to the set $S_p$ i.e. $S_p = S_p \cup q$
7:         else
8:             Increment the domination counter i.e. $n_p = n_p + 1$
9:         end if
10:    end for
11:    if $n_p = 0$ then
12:        No individuals dominate $p$, $p$ belongs to the first front, i.e. $p_{rank} = 1$
13:        Update the first front set by adding $p$ to it. i.e. $F_1 = F_1 \cup p$
14:    end if
15: end for
16: Initialize the front counter to one. $i = 1$
17: while The $i^{th}$ front is nonempty i.e.$F_i \neq \emptyset$ do
18:     Initialize the set for storing the individuals of $(i+1)^{th}$ front to empty. i.e.
        $Q = \emptyset$
19:     for Each individual $p$ in front $F_i$ do
20:         for Each individual $q$ in $S_p$ ($S_p$ denotes the set of individuals dominated by
            $p$) do
21:             Decrement the domination count for individual $q$, i.e. $n_q = n_q - 1$
22:             if $n_q = 0$ then
23:                 No individuals in the subsequent fronts dominate $q$. Set $q_{rank} = i + 1$.
24:                 Update the set $Q$ with individual $q$ i.e. $Q = Q \cup q$.
25:             end if
26:         end for
27:     end for
28:     $i = i + 1$
29:     Set Q is the next front i.e. $F_i = Q$
30: end while
```

2.2.4 Tournament selection

Tournament selection procedure for MOGA is similar to the selection procedure for SOGA. Tournaments which are composed of randomly selected individuals from the pool are formed. A tournament size is usually greater than or equal to 2. The individual with the highest rank from non-dominated sort is selected. If there are more than two individuals with the same rank, the one with larger crowding distance

Algorithm 2 Crowding Distance

```
1: for Each front $F_i$ with $n$ individuals do
2:     Initialize the distance to be zero for all the individuals, i.e. $F_i(d_j) = 0$, where
       $j$ denotes the index of individual.
3:     for Each objective function $m$ do
4:         Sort the individuals in front $F_i$ based on objective $m$, i.e. $I = sort(F_i, m)$
5:         Assign infinite distance to boundary values for each individual in $F_i$, i.e.
           $I(d_1) = \inf$, $I(d_n) = \inf$
6:         for $k$ equals 2 to $(n-1)$ do
7:             $I(d_k) = I(d_k) + \frac{f_m(k+1) - f_m(k-1)}{f_m^{max} - f_m^{min}}$, $f_m(k)$ denotes the fitness value of the $m^{th}$
               objective function of the $k^{th}$ individual in $I$
8:         end for
9:     end for
10: end for
```

will be selected. Tournament selection mimic the survival selection of evolutionary process.

2.2.5 Crossover and Mutation

The crossover and mutation procedures for MOGA are the same as in SOGA. SBX is used for both crossover and mutation.

2.2.6 Environmental selection

For a MOGA, a new pool will be generated after the tournament selection, crossover and mutation. A process called recombination combines the new pool with the original pool. A non-dominated sort would be performed on the combined pool. The process of selecting the next generation pool from the combined pool based on minimizing the rank and maximizing the crowding distance is called environmental selection. For a MOGA, the environmental selection is the dominant selection procedure and poses heavier selection pressure than the tournament selection.

2.2.7 Termination

The MOGA terminates if the maximum iteration occurs. During the MOGA process, the information of chromosomes, fitness values, rankings and crowding distances from each generation is collected. Those data are to be analyzed afterward.

2.3 Optimum Computing Budget Allocation Schemes

A genetic algorithm achieves better accuracy by integrating computer budget allocation schemes into either the selection or evaluation process when solving optimization problems with uncertainty. The procedure of integration into either the selection or evaluation process is easy to understand and will be elaborated in the following chapters. However, allocation schemes are also essential. Developed by Chen et al., optimum computing budget allocations (OCBA/OCBA-m) are sequential ranking and allocation procedures developed from ordinal optimization (OO) [24]. They are integrated with different algorithms and applied to realistic problems [3] [25] [26] [27] [28] and also inside genetic algorithms [6].Although OO could significantly reduce the computational cost with "good-enough" solutions [3] [29], the OCBA procedure further enhances the efficiency and optimally chooses the number of simulations for each design, obtaining maximum computing efficiency with the highest probability of correct selection.

The probability of correct selection or $P(CS)$ is defined to give the probability of the observed good-enough solutions, selected via order comparison, are indeed the true good-enough solutions [30]. The calculation of probability of correct selection is very time-consuming via Monte Carlo simulation [24]; therefore, Chen et al [24] proposed lower bound estimation under the following assumptions: assume the simulation output $L(X_i, \xi_{ij})$ to be a normal distribution with unknown mean, assume the variance σ_i^2 to be known for the purpose of simplicity, and assume a non-informative prior is applied (implies no prior knowledge about any designs). Then, under the Bayesian model, the posterior distribution of J_i is also a normal distribution given by:

$$p(J_i|L(Xi,\xi_{ij}), j = 1, 2, \ldots, N_i) \sim N(\bar{J}_i, \frac{\sigma_i^2}{N_i}) \tag{2.5}$$

in which $\bar{J}_i$ is the sample mean, and variance σ_i^2 is estimated using the sample variance s_i^2.

The definition of $P(CS)$ is:

$$\begin{aligned} P(CS) &= P\{\mathit{design\ b\ is\ actually\ the\ best\ one}\} \\ &= P\{J_b < J_i, \forall i, j = 1, \ldots, N_i, i = 1, \ldots, k, i \neq b\} \end{aligned} \tag{2.6}$$

The two forms of Approximate Probability of Correct Selection (APSCS) derived from statistics literature [31] [32] [33] are described in the following Lemmas [34]:

Lemma 1. The approximate probability of correct selection using the Bonferroni inequality is:

$$\begin{aligned} P(CS) &\geq 1 - \sum_{i=1, i\neq b}^{k} P\{\tilde{J}_b > \tilde{J}_i\} \\ &= 1 - \sum_{i=1, i\neq b}^{k} \Phi\left(\frac{\delta_{b,i}}{\sigma_{b,i}}\right) \\ &= APCS - B \end{aligned} \tag{2.7}$$

Lemma 2. The approximate probability of correct selection using a product form is:

$$\begin{aligned} P(CS) &\geq \prod_{i=1, i\neq b}^{k} P\{\tilde{J}_b < \tilde{J}_i\} \\ &= \prod_{i=1, i\neq b}^{k} \Phi\left(\frac{-\delta_{b,i}}{\sigma_{b,i}}\right) \\ &= APCS - P \end{aligned} \tag{2.8}$$

where $\tilde{J}_i \sim N(\bar{J}_i, \frac{\sigma_i^2}{N_i})$, $\bar{J}_b \leq min_i\ \bar{J}_i$,$\delta_{b,i} = \bar{J}_b - \bar{J}_i$,$\sigma_{b,i}^2 = s_i^2/N_i + s_b^2/N_b$, k is the total number of designs and $\Phi(x)$ is the standard normal cumulative distribution function.

The two methods of calculating APCS are similarly easy and quick without the necessity of Monte Carlo simulation. Chen et al. developed two distinct procedures of OCBA based on these two methods. The classical OCBA scheme is derived by maximizing the APCS-B and allocating samples with a total computing budget of T. The APCS-B is calculated to estimate the probability of correct selection, $P(CS)$. This equation is then used to form a Lagrangian relaxation and to find the stationary point of the Lagrangian function through calculation. The classical OCBA scheme asymptotically maximizes the APCS-B using Lemma 1 with a total computing budget of T and minimizes the total computing cost to maximize the probability of correct selection while choosing the optimum design.

The OCBA-m scheme identifies the optimum subset of m designs out of total of k designs. The probability of correct selection is estimated using Lemma 2:

$$\begin{aligned} P(CS) &= P\{\tilde{J}_i \leq \tilde{J}_j, i \in S_m\ and\ j \notin S_m\} \\ &= P\{\tilde{J}_i \leq c\ and\ \tilde{J}_j \geq c, i \in S_m\ and\ j \notin S_m\} \\ &= \prod_{i \in S_m} P\{\tilde{J}_i \leq c\} \prod_{i \notin S_m} P\{\tilde{J}_i \geq c\} \\ &= APCS - m \end{aligned} \tag{2.9}$$

By calculating the APCS-m and using it to form a Lagrangian relaxation, the stationary point can also be found. Based on these calculations, Chen et al state the asymptotic solution for classical OCBA [24] and OCBA-m [34] in the following two theorems:

Theorem 1. Given a total computing budget (T) to be allocated to k competing designs with performance measured by means $\bar{J}(X_1), \bar{J}(X_2), \ldots, \bar{J}(X_k)$ and finite variances $\sigma_1^2, \sigma_2^2, \ldots, \sigma_k^2$, respectively, as $T \to \infty$, the approximate probability of correct selection can be asymptotically maximized when:

$$\frac{N_i}{N_j} = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_j/\delta_{b,j}}\right)^2, i, j \in \{1, 2, , \dots, k\}, and\ i \neq j \neq b \tag{2.10}$$

$$N_b = \sigma_b \sqrt{\sum_{i=1, i\neq b}^{k} \frac{N_i^2}{\sigma_i^2}} \tag{2.11}$$

Theorem 2. Given a total computing budget (T) to be allocated to k competing designs with performance measured by means $\bar{J}(X_1), \bar{J}(X_2), \dots, \bar{J}(X_k)$ and finite variances $\sigma_1^2, \sigma_2^2, \dots, \sigma_k^2$,respectively, as $T \to \infty$, the approximate probability of correct selection for top m best designs (APCS-m) can be asymptotically maximized when:

$$\frac{N_i}{N_j} = \left(\frac{\sigma_i/\delta_i}{\sigma_j/\delta_j}\right)^2, i, j \in \{1, 2, , \dots, k\}, and\ i \neq j \tag{2.12}$$

Where $\delta_i = \bar{J}_i - c$, $c = \frac{\hat{\sigma}_{i_{m+1}} \bar{J}_{i_m} + \hat{\sigma}_{i_m} \bar{J}_{i_{m+1}}}{\hat{\sigma}_{i_m} + \hat{\sigma}_{i_{m+1}}}$ and, $\hat{\sigma_i} \equiv \sigma_i/\sqrt{N_i}$.

In the above theorems, $\frac{\delta_{b,i}}{\sigma_i}$ from Theorem 1 and $\frac{\delta_i}{\sigma_i}$ from Theorem 2 can be considered as signal-to-noise ratios for design i. In selecting the best design, high $\frac{\delta_{b,i}}{\sigma_i}$ means that either the design i is much worse than the best design, or the design has smaller noise. This condition increases confidence in differentiating this design from the best design; thus fewer samples are needed to allocate further. In the selection of the top m designs, a higher ratio, $\frac{\delta_i}{\sigma_i}$, indicates greater certainty that design i is either included in the top m subset or is among the worst designs; in both cases, fewer samples are needed. When $m = 1$, the OCBA-m selects the best design. Overall, we should spend more samples on the designs which have lower signal-to-noise ratio, and the allocating of computing budget is inversely proportional to the square of the signal to noise ratio.

According to the theorems, the allocation algorithm can be summarized in the following steps:

Input: input number of designs k, initial sample number n_0, and allocation computing budget T. The initial sample number, n_0, should be sufficient to allow sufficient stochastic information while not so large as to allocate excessively many initial samples. Usually, it is chosen according to an estimation of the variance of the noise, defined as the noise level. If the noise level is high, more initial samples should be expected and vice versa for a low noise level. The ratio of $T : n_0$ may vary.

Initialize: initialize n_0 samples for all designs.

Calculate: calculate sample mean $\bar{J}_i = \frac{1}{N_i^l} \sum_{j=1}^{N_i^l} L(X_i, \xi_{ij})$, and sample standard deviation $s_i = \sqrt{\frac{1}{N_i^l - 1} \sum_{j=1}^{N_i^l} (L(X_i, \xi_{ij}) - \bar{J}_i)^2}$, $i = 1, \ldots, k$. Use the updated sample variance s_i^2 as the estimation of variance σ_i^2. For OCBA, find $b = arg\ min_i\ \bar{J}_i$, $i = 1, \ldots, k$, and compute δ_{bi}, $i = 1, \ldots k, i \neq b$. For OCBA-m, use the ranking of $\bar{J}_i$ to determine the subset S_m and compute $\hat{\sigma}_i$, c, and δ_i according to Theorem 2.

Allocate: allocate the computing budget T to k designs. For OCBA, allocate according to Theorem 1. For OCBA-m, allocate according to Theorem 2.

The formulation of OCBA for multi-objective problems [34] follows a similar procedure for the single objective problems illustrated above. The detailed calculation steps for optimal allocation of samples will be included in Chapter 4. Overall, the OCBA schemes aim to maximize simulation efficiency by determining the best numbers of samples for each to achieve the maximum probability of correct selection.

2.4 Three phase Photo-voltaic (PV) inverter

The last topic of this dissertation is about the optimization of a three-phase photovoltaic (PV) inverter system. There has been a major growth in both residential and commercial installation of PV systems recently. The US solar market reported a 45% increase in PV market in 2016 than 2015. Until now, nearly 32 GW of total solar capacity now installed can generate enough electricity to power 6.2 million homes. And the solar prices also dropped a significant of 18% from 2015 to 2016 according to the solar energy industries association reports. The optimization of the three phase PV system had been an important topic in both research and industry due to the increasing demands.

The structure of a three-phase PV inverter system usually contains PV arrays, three-phase inverter, and output filter. A brief introduction to the PV inverter system is given herein.

2.4.1 Photo-voltaic (PV) array

A PV array is a complete power generating unit, which is usually composed of some PV modules or panels. A PV module is one of the fundamental building blocks of PV system, which consists PV cell circuits [35]. PV cells use semiconductor p-n junction to absorb light energy and translate to electrical charge. PV array translates the solar energy into electricity form.

2.4.2 Three phase inverter

The inverter in a PV system is used to transform the DC form of electricity generated from the PV arrays into AC form. The three-phase inverter usually adopts six switches for the three-phase configuration. The six switches are arranged in three parallel legs, with each leg contains two switches in series. The DC input to the three-phase inverter is applied between the top and bottom of the three legs. The three phase output is produced in each leg for each phase using the two series switches. The switching is performed by Pulse-Width Modulation (PWM). PWM is the process of modifying the width of the pulse in a series of pulses according to the control signal produced by the control goal and the real time circuit response.

In three phase inverter, the PWM signal is usually produced by comparison of a sinusoidal control signal and the triangular switching signal [36]. The control signal is traditionally produced out of proportional integral (PI) current control [37]. However, this control strategy is known for several drawbacks: the steady-state errors in single-phase systems and the need for synchronous d-q transformation in three-phase systems. Based on these drawbacks, the proportional-resonant (PR) controller is used in this study. Another main benefit is to implement selective harmonic compensation without requiring excessive computational resource [38]. The PR control is now widely used in grid-interfaced converters.

Another important control loop for grid connected three phase inverter is the phase locked loop (PLL) to provide the rotational frequency and angle by resolving the grid voltage *abc* component. The grid voltage is first transformed to *qd* frame; then a PI controller is used to force the quadrature component of the voltage to zero. The block diagram of the PLL is shown in Figure 2.1

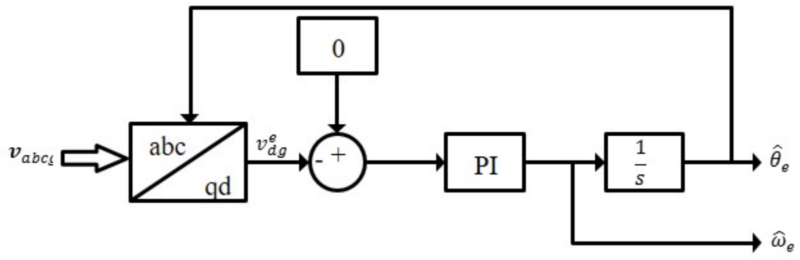

Figure 2.1: Block diagram of the phase locked loop (PLL)

The *abc*/*qd* transformation is used to transform quantities in the synchronous reference frame (*abc*) to a synchronously rotating reference frame (*qde*). The benefits of the translation are that when the system is balanced, the components would be constant. The equations of the transformation are given in Eq. 2.13 and Eq. 2.14.

$$\begin{aligned} \mathbf{f}_{qde} &= \mathbf{K_r}\mathbf{f}_{abc} \\ \mathbf{K_r} &= \frac{2}{3}\begin{bmatrix} \cos\theta & \cos(\theta-\frac{2\pi}{3}) & \cos(\theta+\frac{2\pi}{3}) \\ \sin\theta & \sin(\theta-\frac{2\pi}{3}) & \sin(\theta+\frac{2\pi}{3}) \\ \frac{1}{2} & \frac{1}{2} & \frac{1}{2} \end{bmatrix} \end{aligned} \tag{2.13}$$

$$\mathbf{f}_{abc} = \mathbf{K_r}^{-1}\mathbf{f}_{qde}$$
$$\mathbf{K_r}^{-1} = \begin{bmatrix} \cos\theta & \sin\theta & 1 \\ \cos(\theta - \frac{2\pi}{3}) & \sin(\theta - \frac{2\pi}{3}) & 1 \\ \cos(\theta + \frac{2\pi}{3}) & \sin(\theta + \frac{2\pi}{3}) & 1 \end{bmatrix} \tag{2.14}$$

For a balanced network, after the transformation, Eq. 2.15 can be obtained.

$$v^e_{dge} = \mathbf{v}_s \sin(\omega_e - \hat{\omega}_e) \tag{2.15}$$

The rotational frequency can be obtained by driving v^e_{dge} to zero using PI control:

$$\hat{\omega}_e = k_p(0 - v^e_{dge}) + k_i \int (0 - v^e_{dge})dt \tag{2.16}$$

The rotation angle θ is the integral of radiance:

$$\theta = \int \hat{\omega}_e dt \tag{2.17}$$

2.4.3 Filter

The output quantity from a three-phase inverter contains switching harmonics along with the fundamental desired frequency. A filter plays a critical role in the transaction from the output to the grid by separating the fundamental frequency part from the high-frequency noise. The output filter can be designed to L filter, LC filter or LCL filter according to requirements. Among the three types of filters, LCL is the kind that has the most significant filtering effect. However, because of the complexity of the second order system, the inverter with LCL filter is less stable than the other two kinds. A virtual resistance damping method is used for the grid current controlled inverter to increase the stability of the system. More technical details about the controlling and the damping will be introduced in Chapter 5.

CHAPTER 3
SINGLE-OBJECTIVE GENETIC ALGORITHM WITH ALLOCATION TECHNIQUES FOR STOCHASTIC PROBLEMS

Stochastic problems are widely encountered in almost every field of science and engineering. Many realistic industrial problems are subject to randomness or to uncertainties that can be modeled stochastically. Example problems include stochastic scheduling [11,39], computation of optimal power flow and optimization of an operating point with constraints [40], stochastic network partitioning and clustering [41,42], stochastic vehicle routing problem [43], and design of power system stabilizers [44]. A common feature of these example problems is the relatively high computational expense of performing the required simulations. The straightforward approach of collecting a large number of samples can be computationally very expensive. Improvements to this approach can be made by use of efficient sampling in the search algorithm for these types of problems, potentially improving both the runtime and the accuracy of the search algorithm.

Meta-heuristic methods, such as genetic algorithms (GAs), particle swarm optimization, simulated annealing and ant colony optimization, have been widely used to solve optimization problems [45, 46], and they have been particularly successful when the derivative of the objective function is unknown or does not exist. However, when randomness is introduced into these problems, appropriate methods for addressing it are necessary. Some methods seek a robust solution, which is insensitive to small variations [47], such as analog circuits designed to be robust against certain faults [48]. Variations of the heuristic algorithms have been proposed to find solutions to stochastic problems. Such variations include specifying the probability of constraint satisfaction as part of the fitness function of a GA [49], introducing a probability vector in the extension of each colony for ant colony optimization [50], integrating statistical sequential selection into simulated annealing [51], incorporating equal re-sampling methods into particle swarm optimization [52], integrating mathematical approximations of a solution's reliability into evolutionary algorithms and investigating the situation case by case [53], employing the concept of global and local optimization and improving the efficiency of globally robust search by dividing the search space into regions [54], and by constructing local approximate models of the fitness function [55]. These approaches have primarily focused on the quality of the solutions, with only secondary concern for computational efficiency.

Applying ordinal optimization techniques, which seek to allocate adequate numbers of samples to promising individuals and reduce unnecessary sampling of non-critical individuals, is a promising approach to improve efficiency while maintaining accuracy at locating the solution to stochastic problems [56,57]. In this direction, the optimal computing budget allocation (OCBA) methods, which are sequential ranking and allocation procedures developed from ordinal optimization [24], have been developed. OCBA methods and other methods for computing budget allocation (CBA) are used to allocate the samples that are performed on a set of individuals to reveal the relative merit of the individuals. Combining CBA methods into heuristic search algorithms, such as GAs, is a good way of improving the sampling efficiency in robust optimization methods. A technique for integrating statistical ranking into evolutionary algorithms is presented in [6]. The OCBA method is proposed to guide the allocation of sampling budget and identification of good particles in particle swarm optimization [7], and it is further developed and investigated with various distributions in [58]. The previous usage of CBA methods with evolutionary algorithms has focused on the integration of the CBA methods into the evaluation and ranking process [3, 6, 59]. In [60], the OCBA rule for selecting the best m individuals out of a set of k is integrated with GA evaluation to improve the search efficiency. A GA involving CBA methods in this way is called an evaluation-integrated GA (EIGA).

The contribution of this work is to propose a selection-integrated GA (SIGA) in which CBA techniques are integrated directly into the GA selection operator rather than being used during fitness evaluation. The SIGA allows fitness evaluations to be allocated towards specific individuals for whom the GA requires more information. Several stochastic test problems are considered under different noise levels, including problems based on benchmark functions from a recent conference competition, and the performance of the EIGA and the SIGA with different CBA methods is compared. Statistical significance tests are performed on those problems to verify the accuracy of the proposed algorithm. It is shown that the SIGA is capable of achieving more accurate results for the same computational budget or results with the same accuracy for a considerably decreased computing budget.

The remainder of this chapter is organized as follows. First, descriptions of the problem and different CBA methods are given. Then, the existing EIGA and proposed SIGA approaches for integrating CBA methods into GAs are described. Next, test functions, including various noise levels, are presented, and experimental results and analysis are given.

3.1 Stochastic problems and computing budget allocation methods

The stochastic problems and the CBA methods considered herein are described below.

3.1.1 Stochastic problem statement

The stochastic problems considered herein have the following form [61]:

$$\min_X \; J(X) = E[L(X, \xi)], \tag{3.1}$$

where X is the (possibly multi-dimensional) decision variable, $L(\cdot,\cdot)$ is the sample performance and can only be calculated via simulation, ξ is a random variable representing the noise integrated within the function, and $J(\cdot)$ represents the expected performance. For the test problems discussed herein, the randomness is modeled as additive noise with a Gaussian distribution, but it can be represented in other ways as well. It is also assumed that such a problem is unconstrained in the sense that any constraints that bind the solution are appropriately penalized in $L(\cdot,\cdot)$.

For a deterministic problem $J(X) = L(X)$, GAs are widely applied to find the optimal solution $X^* = \arg\min_X L(X)$. For a stochastic problem, the fitness function $J(X)$ can only be estimated by calculating the mean of a limited number of random samples. For individual i, the mean performance measure can be estimated as

$$J(X_i) \approx \bar{L}_i = \frac{1}{n_i} \sum_{j=1}^{n_i} L(X_i, \xi_{ij}), \tag{3.2}$$

where n_i represents the number of samples and ξ_{ij} represents the noise of the jth sample for individual i. The variance σ_i^2 for individual i is unknown and can be approximated by the sample variance s_i^2; the sample mean for individual i is denoted as $\bar{L}_i$; the optimum individual given previous samples is denoted as b, which has the smallest sample mean, i.e. $\bar{L}_b \leq \bar{L}_i, \forall i$.

Due to the law of large numbers, increasing n_i will result in $\bar{L}_i$ being a better estimate of the actual mean $J(X_i)$. However, evaluating more samples requires more computational time. GAs, depending on the fitness scaling and selection methods used, often require an understanding of the relative fitness of smaller sets of individuals rather than a precise understanding of the absolute fitness of each. The goal of the proposed SIGA is to provide this information to the GA. Under the cases of two individuals have sufficiently different sample means or small sample variances such that one individual is very likely better than the other, there is no need to evaluate more samples of these individuals.

3.1.2 Computing budget allocation methods

Various CBA methods exist in the literature, and the CBA methods applied in this study are discussed in this subsection. For each of these methods, it is assumed that N samples are being allocated among k individuals.

Equal allocation method

The simplest allocation technique to conduct sampling is the equal allocation (EQU) technique and it often serves as a benchmark for comparison [34]. The available computing budget is equally distributed to all individuals being compared:

$$\tilde{n}_i = \frac{N}{k}, \tag{3.3}$$

where $\tilde{n}_i$ is the number of additional samples to be allocated to individual i.

Optimal computing budget allocation methods

OCBA methods are based on asymptotic arguments that show that allocating samples in a particular manner will result in the highest probability of correct selection. The two OCBA methods used herein, described in [12], are based on different assumptions and are treated separately. In both OCBA methods, some initial samples n_i have been performed for each individual, and information about the sample mean

$\bar{L}_i$ and variance σ_i^2 of each individual is used to perform the allocation. The available N samples are allocated to individuals based on assumptions about distribution and asymptotic behavior to maximize the probability of correctly selecting the best individual (OCBA) or selecting the best m individual subset (OCBAM). In this study, $m = 1$, and the OCBAM represents an alternative OCBA method of correctly selecting the best individual.

For the OCBA method, the best individual is identified:

$$b = \arg\min_i \bar{L}_i, \tag{3.4}$$

and this individual is used to calculate a distance

$$\delta_{b,i} = \bar{L}_i - \bar{L}_b. \tag{3.5}$$

This distance and the sample variance are used to establish the ratio of samples allocated to different individuals:

$$\frac{\tilde{n}_i}{\tilde{n}_j} = \left(\frac{\sigma_i/\delta_{b,i}}{\sigma_j/\delta_{b,j}}\right)^2, \; i \neq b, \; j \neq b \tag{3.6}$$

The number of samples allocated to the best individual is given by

$$\tilde{n}_b = \sigma_b \sqrt{\sum_{i=1,\; i\neq b}^{k} \frac{\tilde{n}_i^2}{\sigma_i^2}}, \tag{3.7}$$

and the total number of allocated samples must be N.

For the OCBAM method, the individuals are sorted such that i is the rank of the individual (i.e., the first individual is the best individual). A boundary between the best m individual subset and the remainder of the set is established as

$$c = \frac{\hat{\sigma}_{m+1}\bar{L}_m + \hat{\sigma}_m \bar{L}_{m+1}}{\hat{\sigma}_m + \hat{\sigma}_{m+1}}, \tag{3.8}$$

where $\hat{\sigma_i} = \sigma_i/\sqrt{n_i}$. This boundary is used to calculate a distance

$$\delta_i = \bar{L}_i - c. \tag{3.9}$$

This distance and the sample variance are used to establish the ratio of samples allocated to different individuals:

$$\frac{\tilde{n}_i}{\tilde{n}_j} = \left(\frac{\sigma_i/\delta_i}{\sigma_j/\delta_j}\right)^2, \tag{3.10}$$

and the total number of allocated samples must be N.

Proportional to variance method

Like the OCBA methods, the proportional to variance (PTV) method uses information about the initial n_i samples of each individual (i.e., the variance σ_i^2) to perform the allocation. The PTV method allocates budget in proportion to the variances because a smaller calculated sample variance usually implies more certainty [62]. The allocation is given by

$$\frac{\tilde{n}_i}{\tilde{n}_j} = \frac{\sigma_i^2}{\sigma_j^2}, \tag{3.11}$$

and the total number of allocated samples must be N.

Algorithm 3 EIGA: sigle objective

```
1:  Initialize population of individuals with random values in search space
2:  Evaluate each individual in initial population with number of initial samples
3:  if EIGA then
4:      Apply CBA to allocate additional samples for each individual in initial population
5:  end if
6:  while Maximum generation not reached do
7:      for Each tournament do
8:          Select the best individual from tournament into mating pool
9:      end for
10:     Perform crossover for each pair of individuals in mating pool
11:     Perform mutation on each individual in mating pool
12:     Evaluate each individual in mating pool with number of initial samples
13:     if EIGA then
14:         Apply CBA to allocate additional samples for each individual in mating pool
15:     end if
16:     Insert mating pool into population
17: end while
```

3.2 Computing budget allocation method integrated genetic algorithms

Herein, two fundamental GAs for solving optimization of stochastic problems are considered. The first algorithm is the EIGA and is the traditional approach for integrating CBA schemes with GAs. The second algorithm is the SIGA and is proposed herein as an alternative method for solving stochastic problems. Figure 3.1 gives the flow chart of the integrated GAs for EIGA, and Figure 3.2 gives the flow chart of the integrated GAs for SIGA. From Figure 3.1, the EIGA integrates the CBA procedure into the evaluation process. From Figure 3.2, the SIGA integrates the CBA procedure after the evaluation process, at the point in which order information is required by the GA. Therefore, it includes an additional application of the CBA method after termination to determine the final solution.

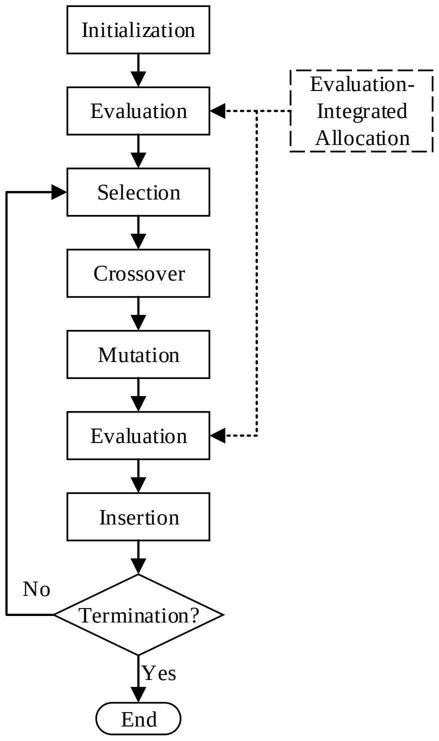

Figure 3.1: Flow chart of the evaluation-integrated genetic algorithm

Pseudo-codes for the two algorithms are listed in Algorithm 3 and Algorithm 4, and steps specific for each algorithm are marked in italics. The integrated GAs have the following steps:

Step 1: The population of n_{ind} individuals is uniformly randomly initialized in the search domain.

Step 2: Each individual is initially sampled n_0 times, and the sample mean and variance of each individual are calculated. In the EIGA, the CBA method uses the sample means and variances to allocate $n_1 n_{ind}$ additional samples across the population.

Step 3: Tournament selection is used. The tournament size is n_{tour} with one individual selected from each tournament, and n_{pool} tournaments are used to form the mating pool. For the EIGA, the selection is performed using the previously updated

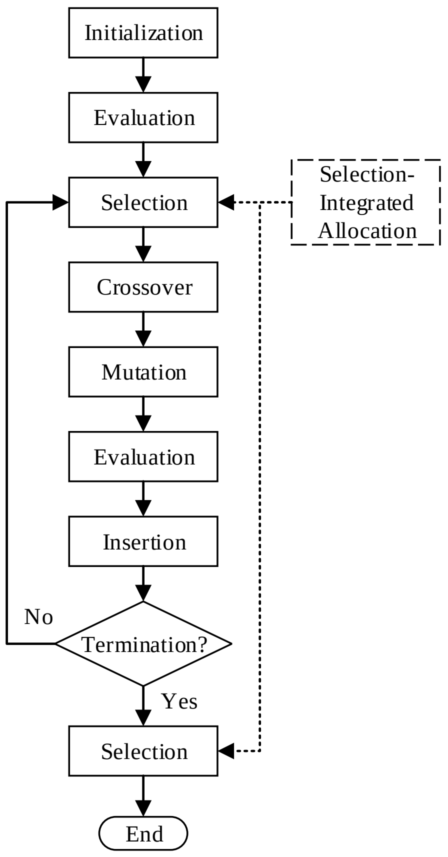

Figure 3.2: Flow chart of the selection-integrated genetic algorithm

sample means. For the SIGA, n_1 additional samples are allocated to each tournament by the CBA method, and the tournament winner is decided after these samples are performed. The tournaments are processed sequentially such that samples allocated to an individual in one tournament are considered if the individual participates in subsequent tournaments.

Step 4: Individuals in the mating pool are arranged in pairs, and a crossover is performed on each pair. The simulated binary crossover (SBX) method [17, 18] is used with crossover constant η.

Step 5: Polynomial mutation for real-valued GAs [63] is performed on each individual in the mating pool with probability p_m.

Algorithm 4 SIGA: single objective

```
 1: Initialize population of individuals with random values in search space
 2: Evaluate each individual in initial population with number of initial samples
 3: while Maximum generation not reached do
 4:     if SIGA then
 5:         for Each tournament do
 6:             Apply CBA to allocate additional samples for each individual in tournament
 7:         end for
 8:     end if
 9:     for Each tournament do
10:         Select best individual from tournament into mating pool
11:     end for
12:     Perform crossover for each pair of individuals in mating pool
13:     Perform mutation on each individual in mating pool
14:     Evaluate each individual in mating pool with number of initial samples
15:     Insert mating pool into population
16: end while
17: if SIGA then
18:     Apply CBA to allocate additional samples for each individual in final population
19: end if
```

Step 6: Each individual in the mating pool is initially sampled n_0 times, and the sample mean and variance are of each individual are calculated. In the EIGA, the CBA method uses the sample means and variances to allocate $n_1 n_{pool}$ additional samples across the mating pool.

Step 7: The new n_{pool} individuals randomly replace n_{pool} individuals in the population. Elitism is not used as the fitness evaluation is uncertain due to the noise.

Step 8: The algorithm terminates after n_{gen} generation. After the SIGA completes its final generation, the CBA method is used to allocate $n_1 n_{ind}$ samples across the final population to identify the best individual.

The total samples required by each method is equal to $n(n_{ind} + n_{gen} n_{pool})$, where $n = n_0 + n_1$ is the total number of samples to be performed per individual per generation.

The primary focus of this work is to compare EIGA with SIGA. The particular genetic operators employed by these algorithms or their parameters are considered to be secondary to the manner in which CBA is performed. The parameters for the GAs are given in Table 3.1. While the parameters can certainly affect the performance of the GA, the selection of these parameters is not the focus of this work. Therefore, the parameters are selected manually to achieve acceptable performance on the deterministic versions of the test functions described below and used for both the EIGA and the SIGA. It should also be noted that elitism is not used by any of the algorithms. While elitism has been shown near universally to improve the performance of GAs, the stochastic problems pose particular difficulties for the use of elitism. In particular, knowledge of whether an individual is elite is based on an imperfect sample mean. Elitism combined with a particularly unlucky sample can derail the algorithm.

Table 3.1: Parameters of genetic algorithms

Parameter	Parameter	Value
n_{gen}	Number of generations	300
n_{ind}	Number of individuals	100
n_{pool}	Number of individuals in mating pool	60
n_{tour}	Tournament size	4
η	Crossover constant	2
p_c	Crossover probability	100%
p_m	Mutation probability	0.05
n_0	Initial samples per individual	50
n_1	Allocated samples per individual	200

3.3 Results and analysis

This section presents several stochastic test problems, describes experimental results, and compares the EIGA and SIGA as well as the CBA methods.

3.3.1 Test functions

Several deterministic test functions with known global minima are selected. Zero-mean Gaussian noise is added to each deterministic test function in order to form a stochastic test function with known global expected minima:

$$L(X, \xi) = f(X) + \xi \tag{3.12}$$

$$\xi = \mathcal{N}(0, D^2) \tag{3.13}$$

where D is a parameter establishing the noise level associated with the problem. The global expected minimizer of the stochastic problem is equal to the global minimizer of the deterministic problem:

$$X^* = \arg\min_X E[L(X, \xi)] = \arg\min_X f(X), \tag{3.14}$$

and the global expected minimium is equal to the global minimum of the deteministic problem:

$$E[L(X^*, \xi)] = f(X^*). \tag{3.15}$$

In order to evaluate the quality of a solution X proposed by one of the optimization methods, its error with respect to the global minimum is computed:

$$\Delta = f(X) - f(X^*), \tag{3.16}$$

which is possible for the test functions because the global minimum is known.

The algorithms are tested on four traditional test functions and three functions based on benchmark functions from a recent conference competition. The plate-shaped Matyas function has one global optimum and no local optima. The valley-shaped Rosenbrock function is one of the most popular optimization test problems. Its global optima are situated inside a parabolic valley that makes convergence difficulty. The two higher dimensional test functions are the bowl-shaped Sphere function and the plate-shaped Zakharov function. The traditional test functions are listed in Table 3.2. Three test functions from the 2014 IEEE Congress on Evolutionary Computation competition are used [64]. The three functions used are the Griewank, HappyCat and HGBat functions, and these functions are shifted and rotated. The dimensions used for these functions are $d = 10$ for the Griewank function and $d = 30$ for the both the HappyCat and HGBat functions. The search domain for these functions is $[-100, 100]^d$. It is noted that these problems are intended to be representative of modern benchmark functions but that not all functions that are appropriate benchmark functions for deterministic optimization are suitable stochastic problems.

Table 3.2: Traditional test functions

Function	d	Definition	Global minimum	Search domain
Matyas	2	$f(X) = 0.26(x_1^2 + x_2^2) - 0.48x_1x_2$	$f(0,0) = 0$	$x_i \in [-10, 10]$
Rosenbrock	2	$f(X) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2$	$f(1,1) = 0$	$x_i \in [-2.048, 2.048]$
Sphere	8	$f(X) = \sum_{i=1}^{d} x_i^2$	$f(0,\ldots,0) = 0$	$x_i \in [-5, 5]$
Zakharov	5	$f(X) = \sum_{i=1}^{d} x_i^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{d} 0.5ix_i\right)^4$	$f(0,\ldots,0) = 0$	$x_i \in [-5, 5]$

3.3.2 Choice of noise level

To determine proper noise levels for each of the test functions, a traditional GA with the same genetic operators and parameters is applied to the deterministic test functions. The traditional GA is a special case of both the EIGA and the SIGA when $n_0 = 1$ and $n_1 = 0$. This GA was applied 1600 times for each allocation method with each problem and noise level, and the mean error and the standard deviation (STD) of the error for each case (when $D = 0$) are given in Table 3.3 and Table 3.4.

The choice of the noise level D is related to the performance of the traditional GA on the deterministic test problems. Using the EIGA with EQU (EQUE) as a benchmark, the noise levels were selected so that the mean errors of the EQUE when executed 1600 times were within the range of 1–20 times the mean error of the traditional GA on the deterministic test problems. Three different noise levels in the range were compared to measure the performance of the GAs under relatively small, moderate, and high noise conditions. The performance of the benchmark EQUE under these noise levels is also shown in Table 3.3 and Table 3.4.

Table 3.3: Mean and standard deviation (STD) of error under different noise levels (part 1)

Function	D	Mean	STD
Matyas	0	4.59×10^{-5}	1.12×10^{-4}
	0.01	1.74×10^{-4}	3.04×10^{-4}
	0.05	4.89×10^{-4}	7.01×10^{-4}
	0.1	7.64×10^{-4}	1.02×10^{-3}
Rosenbrock	0	1.41×10^{-2}	2.78×10^{-2}
	1	3.89×10^{-2}	5.27×10^{-2}
	5	7.46×10^{-2}	8.83×10^{-2}
	10	1.10×10^{-1}	1.17×10^{-1}
Sphere	0	4.38×10^{-3}	4.08×10^{-3}
	0.1	7.86×10^{-3}	5.27×10^{-3}
	0.5	1.69×10^{-2}	9.99×10^{-3}
	2	3.68×10^{-2}	1.98×10^{-2}
Zakharov	0	5.17×10^{-3}	5.95×10^{-3}
	0.1	8.73×10^{-3}	7.35×10^{-3}
	1	2.54×10^{-2}	2.07×10^{-2}
	2	3.82×10^{-2}	2.93×10^{-2}

Table 3.4: Mean and standard deviation (STD) of error under different noise levels (part 2)

Function	D	Mean	STD
Griewank	0	1.13	2.82×10^{-1}
	50	2.14	5.55×10^{-1}
	250	5.18	2.16
	500	8.90	4.23
HappyCat	0	5.09×10^{-1}	1.27×10^{-1}
	10	1.57	7.82×10^{-1}
	50	4.55	7.54×10^{-1}
	100	5.99	1.03
HGBat	0	1.08	1.85
	10	2.14	2.81
	50	7.46	4.78
	100	1.27×10^{1}	5.24

3.3.3 Comparison of different strategies

Experiments were performed to compare the EIGA and SIGA with different CBA methods under the various noise levels identified in Table 3.3 and Table 3.4. The EIGA and SIGA with different CBA schemes are denoted as EQUE, EQUS, OCBAE, OCBAS, OCBAME, OCBAMS, PTVE, PTVS, with the last letter 'E' or 'S' denoting

Table 3.5: Detailed nomenclature of EIGA and SIGA with different CBA schemes

Name	Method	Name	Method
EQUE	EIGA with EQU	EQUS	SIGA with EQU
OCBAE	EIGA with OCBA	OCBAS	SIGA with OCBA
OCBAME	EIGA with OCBAM	OCBAMS	SIGA with OCBAM
PTVE	EIGA with PTV	PTVS	SIGA with PTV

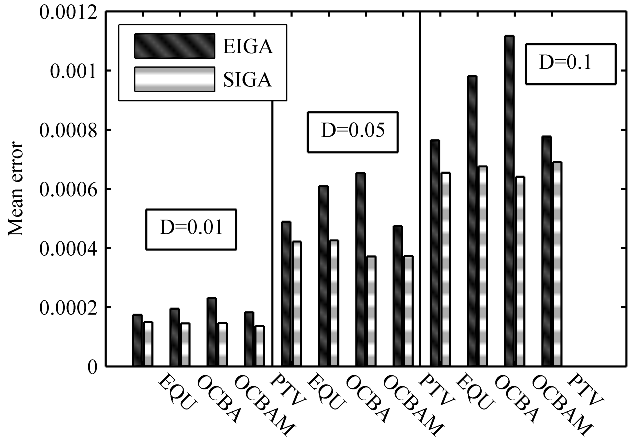

Figure 3.3: Comparison of mean error for Matyas function

the EIGA or SIGA, respectively. Detailed nomenclature can be seen in Table 3.5. Each algorithm was applied to each problem with each noise level using each CBA method 1600 times. Table 3.6 and Table 3.7 show the mean error for each method. The best algorithm for each problem and the noise level is indicated with boldfaced text. Examples of visualization of the table data are shown in Figure 3.3 to 3.6.

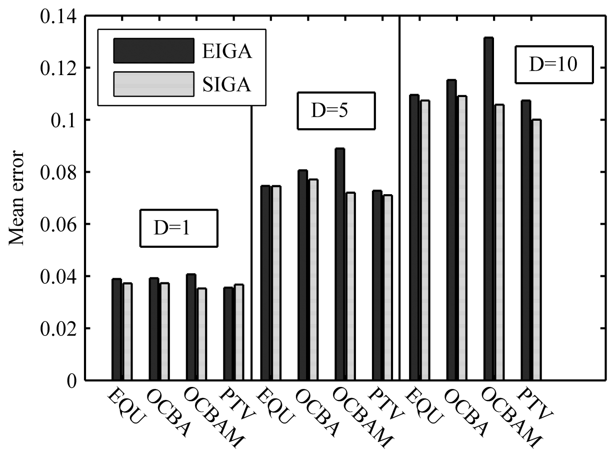

Figure 3.4: Comparison of mean error for Rosenbrock function

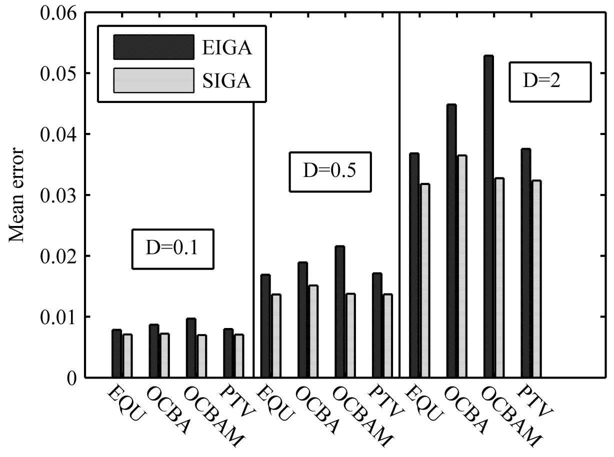

Figure 3.5: Comparison of mean error for Sphere function

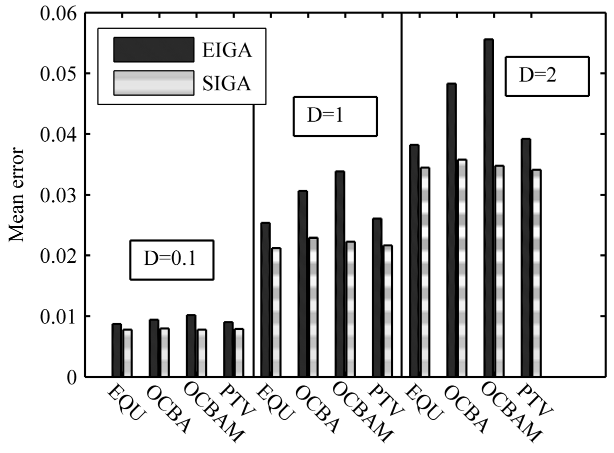

Figure 3.6: Comparison of mean error for Zakharov function

Table 3.6: Part 1 of comparison of mean error for test functions (bold-face text indicates the best algorithm for each problem and noise level)

Function	Method	Mean	Method	Mean	Method	Mean	Method	Mean
Matyas	EQUE	1.74×10^{-4}	OCBAE	1.95×10^{-4}	OCBAME	2.30×10^{-4}	PTVE	1.83×10^{-4}
$D = 0.01$	EQUS	1.50×10^{-4}	OCBAS	1.46×10^{-4}	OCBAMS	1.47×10^{-4}	**PTVS**	$\mathbf{1.37 \times 10^{-4}}$
Matyas	EQUE	4.89×10^{-4}	OCBAE	6.09×10^{-4}	OCBAME	6.54×10^{-4}	PTVE	4.75×10^{-4}
$D = 0.05$	EQUS	4.22×10^{-4}	OCBAS	4.26×10^{-4}	**OCBAMS**	$\mathbf{3.72 \times 10^{-4}}$	PTVS	3.73×10^{-4}
Matyas	EQUE	7.64×10^{-4}	OCBAE	9.81×10^{-4}	OCBAME	1.12×10^{-3}	PTVE	7.77×10^{-4}
$D = 0.1$	EQUS	6.55×10^{-4}	OCBAS	6.76×10^{-4}	**OCBAMS**	$\mathbf{6.41 \times 10^{-4}}$	PTVS	6.90×10^{-4}
Rosenbrock	EQUE	3.89×10^{-2}	OCBAE	3.92×10^{-2}	OCBAME	4.07×10^{-2}	PTVE	3.56×10^{-2}
$D = 1$	EQUS	3.72×10^{-2}	OCBAS	3.73×10^{-2}	**OCBAMS**	$\mathbf{3.52 \times 10^{-2}}$	PTVS	3.68×10^{-2}
Rosenbrock	EQUE	7.46×10^{-2}	OCBAE	8.06×10^{-2}	OCBAME	8.90×10^{-2}	PTVE	7.28×10^{-2}
$D = 5$	EQUS	7.45×10^{-2}	OCBAS	7.71×10^{-2}	OCBAMS	7.20×10^{-2}	**PTVS**	$\mathbf{7.11 \times 10^{-2}}$
Rosenbrock	EQUE	1.10×10^{-1}	OCBAE	1.15×10^{-1}	OCBAME	1.32×10^{-1}	PTVE	1.07×10^{-1}
$D = 10$	EQUS	1.07×10^{-1}	OCBAS	1.09×10^{-1}	OCBAMS	1.06×10^{-1}	**PTVS**	$\mathbf{1.00 \times 10^{-1}}$
Sphere	EQUE	7.86×10^{-3}	OCBAE	8.70×10^{-3}	OCBAME	9.70×10^{-3}	PTVE	7.98×10^{-3}
$D = 0.1$	EQUS	7.09×10^{-3}	OCBAS	7.22×10^{-3}	**OCBAMS**	$\mathbf{6.99 \times 10^{-3}}$	PTVS	7.06×10^{-3}
Sphere	EQUE	1.69×10^{-2}	OCBAE	1.89×10^{-2}	OCBAME	2.16×10^{-2}	PTVE	1.71×10^{-2}
$D = 0.5$	**EQUS**	$\mathbf{1.37 \times 10^{-2}}$	OCBAS	1.51×10^{-2}	OCBAMS	1.38×10^{-2}	PTVS	1.37×10^{-2}
Sphere	EQUE	3.68×10^{-2}	OCBAE	4.48×10^{-2}	OCBAME	5.29×10^{-2}	PTVE	3.76×10^{-2}
$D = 2$	**EQUS**	$\mathbf{3.18 \times 10^{-2}}$	OCBAS	3.65×10^{-2}	OCBAMS	3.28×10^{-2}	PTVS	3.24×10^{-2}

Table 3.7: Part 2 of comparison of mean error for test functions (bold-face text indicates the best algorithm for each problem and noise level)

Function	Method	Mean	Method	Mean	Method	Mean	Method	Mean
Zakharov	EQUE	8.73×10^{-3}	OCBAE	9.41×10^{-3}	OCBAME	1.02×10^{-2}	PTVE	9.01×10^{-3}
$D = 0.1$	EQUS	7.79×10^{-3}	OCBAS	7.97×10^{-3}	**OCBAMS**	$\mathbf{7.77 \times 10^{-3}}$	PTVS	7.90×10^{-3}
Zakharov	EQUE	2.54×10^{-2}	OCBAE	3.06×10^{-2}	OCBAME	3.38×10^{-2}	PTVE	2.61×10^{-2}
$D = 1$	**EQUS**	$\mathbf{2.12 \times 10^{-2}}$	OCBAS	2.29×10^{-2}	OCBAMS	2.23×10^{-2}	PTVS	2.16×10^{-2}
Zakharov	EQUE	3.82×10^{-2}	OCBAE	4.83×10^{-2}	OCBAME	5.56×10^{-2}	PTVE	3.92×10^{-2}
$D = 2$	EQUS	3.45×10^{-2}	OCBAS	3.58×10^{-2}	OCBAMS	3.48×10^{-2}	**PTVS**	$\mathbf{3.41 \times 10^{-2}}$
Griewank	EQUE	2.14	OCBAE	2.42	OCBAME	2.64	PTVE	2.16
$D = 50$	EQUS	2.02	OCBAS	2.12	OCBAMS	2.04	**PTVS**	**2.02**
Griewank	EQUE	5.18	OCBAE	6.67	OCBAME	7.79	PTVE	5.32
$D = 250$	EQUS	5.12	OCBAS	5.56	OCBAMS	5.27	**PTVS**	**5.10**
Griewank	EQUE	8.90	OCBAE	1.20×10^{1}	OCBAME	1.39×10^{1}	PTVE	8.97
$D = 500$	**EQUS**	**8.76**	OCBAS	1.02×10^{1}	OCBAMS	9.51	PTVS	9.09
HappyCat	EQUE	1.57	OCBAE	2.10	OCBAME	2.47	PTVE	1.59
$D = 10$	**EQUS**	**1.40**	OCBAS	1.63	OCBAMS	1.48	PTVS	1.43
HappyCat	EQUE	4.55	OCBAE	5.16	OCBAME	5.60	PTVE	4.59
$D = 50$	EQUS	4.44	OCBAS	4.66	OCBAMS	4.43	**PTVS**	**4.40**
HappyCat	EQUE	5.99	OCBAE	6.87	OCBAME	7.34	PTVE	6.07
$D = 100$	EQUS	5.93	OCBAS	6.20	OCBAMS	5.94	**PTVS**	**5.92**
HGBat	EQUE	2.14	OCBAE	2.51	OCBAME	2.84	PTVE	2.27
$D = 10$	**EQUS**	**1.99**	OCBAS	2.24	OCBAMS	2.16	PTVS	2.13
HGBat	EQUE	7.46	OCBAE	9.32	OCBAME	1.08×10^{1}	PTVE	7.74
$D = 50$	EQUS	6.92	OCBAS	7.24	OCBAMS	6.84	**PTVS**	**6.73**
HGBat	EQUE	1.27×10^{1}	OCBAE	1.58×10^{1}	OCBAME	1.76×10^{1}	PTVE	1.30×10^{1}
$D = 100$	EQUS	1.16×10^{1}	OCBAS	1.24×10^{1}	**OCBAMS**	$\mathbf{1.15 \times 10^{1}}$	PTVS	1.18×10^{1}

Comparison of EIGA and SIGA

By examining Table 4.2 vertically, it can be seen that the SIGA with a given CBA method generally outperforms the EIGA with the same CBA method. The errors are assumed to follow a Rayleigh distribution, so an F-test can be used determine the statistical significance of the results [65]. The null hypothesis is that the mean error of the SIGA for a given CBA method is less than or equal to the mean error of the EIGA for the same CBA method. A significance level of $\alpha = 5\%$ is used to determine whether this hypothesis can be rejected. The results of these statistical tests are shown in Table 3.8. Due to a large number of samples (1600), many of the calculated p-values are smaller than the machine epsilon, and these are denoted in the table as '$< \epsilon$'. Statistically, significant results are indicated with boldfaced text. It can be seen that in most cases, the SIGA has statistically significantly less error than the EIGA for a given CBA method. The SIGA only has a larger sample mean error than the EIGA in two cases out of 84 scenarios: the Rosenbrock function with $D = 1$ and PTV and the Griewank function with $D = 500$ and PTV, and not statistically significantly so in either case. Generally, it can be concluded that by integrating the CBA method into the selection process of the SIGA, the search algorithm allocates samples as needed to make the comparisons between individuals that are required by the algorithm. The relatively small size of each tournament for the size of the mating pool may also improve the accuracy of the sampling.

Comparison of computing budget allocation methods

By examining Table 4.2 horizontally, comparisons among the CBA methods can be made. For EIGA, the EQU and PTV methods generally have the smallest error. There are large differences in the error when applying these methods compared to the application of the OCBA methods. For the SIGA, there are relatively small

Table 3.8: P-value of comparison between SIGA over EIGA for each CBA method ($< \epsilon$ indicates p-value less than machine epsilon, bold-face text indicates statistical significance with significance level $\alpha = 0.05$)

Function	D	EQU	OCBA	OCBAM	PTV
Matyas	0.01	$\mathbf{1.29 \times 10^{-5}}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$\mathbf{1.11 \times 10^{-16}}$
	0.05	$\mathbf{1.56 \times 10^{-5}}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$\mathbf{6.46 \times 10^{-12}}$
	0.1	$\mathbf{6.42 \times 10^{-6}}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$\mathbf{3.94 \times 10^{-4}}$
Rosenbrock	1	1.07×10^{-1}	8.04×10^{-2}	$\mathbf{2.49 \times 10^{-5}}$	8.24×10^{-1}
	5	4.88×10^{-1}	1.06×10^{-1}	$\mathbf{1.22 \times 10^{-9}}$	2.50×10^{-1}
	10	2.88×10^{-1}	5.93×10^{-2}	$\mathbf{3.72 \times 10^{-10}}$	$\mathbf{2.36 \times 10^{-2}}$
Sphere	0.1	$\mathbf{1.85 \times 10^{-3}}$	$\mathbf{6.59 \times 10^{-8}}$	$< \boldsymbol{\epsilon}$	$\mathbf{2.60 \times 10^{-4}}$
	0.5	$\mathbf{1.18 \times 10^{-9}}$	$\mathbf{1.52 \times 10^{-10}}$	$< \boldsymbol{\epsilon}$	$\mathbf{9.30 \times 10^{-11}}$
	2	$\mathbf{1.71 \times 10^{-5}}$	$\mathbf{2.94 \times 10^{-9}}$	$< \boldsymbol{\epsilon}$	$\mathbf{1.26 \times 10^{-5}}$
Zakharov	0.1	$\mathbf{6.21 \times 10^{-4}}$	$\mathbf{1.31 \times 10^{-6}}$	$\mathbf{1.15 \times 10^{-14}}$	$\mathbf{9.97 \times 10^{-5}}$
	1	$\mathbf{1.98 \times 10^{-7}}$	$\mathbf{1.1 \times 10^{-16}}$	$< \boldsymbol{\epsilon}$	$\mathbf{6.28 \times 10^{-8}}$
	2	$\mathbf{1.72 \times 10^{-3}}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$\mathbf{4.55 \times 10^{-5}}$
Griewank	50	5.95×10^{-2}	$\mathbf{9.34 \times 10^{-5}}$	$\mathbf{9.79 \times 10^{-14}}$	$\mathbf{3.60 \times 10^{-2}}$
	250	3.62×10^{-1}	$\mathbf{1.42 \times 10^{-7}}$	$< \boldsymbol{\epsilon}$	1.12×10^{-1}
	500	3.28×10^{-1}	$\mathbf{1.04 \times 10^{-6}}$	$< \boldsymbol{\epsilon}$	3.77×10^{-1}
HappyCat	10	$\mathbf{5.00 \times 10^{-4}}$	$\mathbf{2.97 \times 10^{-13}}$	$< \boldsymbol{\epsilon}$	$\mathbf{1.68 \times 10^{-3}}$
	50	2.46×10^{-1}	$\mathbf{1.80 \times 10^{-3}}$	$\mathbf{1.77 \times 10^{-11}}$	1.17×10^{-1}
	100	3.78×10^{-1}	$\mathbf{1.83 \times 10^{-3}}$	$\mathbf{1.17 \times 10^{-9}}$	2.42×10^{-1}
HGBat	10	$\mathbf{2.33 \times 10^{-2}}$	$\mathbf{6.21 \times 10^{-4}}$	$\mathbf{9.10 \times 10^{-15}}$	$\mathbf{3.38 \times 10^{-2}}$
	50	$\mathbf{1.79 \times 10^{-2}}$	$\mathbf{5.32 \times 10^{-13}}$	$< \boldsymbol{\epsilon}$	$\mathbf{3.78 \times 10^{-5}}$
	100	$\mathbf{7.22 \times 10^{-3}}$	$\mathbf{9.34 \times 10^{-12}}$	$< \boldsymbol{\epsilon}$	$\mathbf{4.08 \times 10^{-3}}$

differences in the error regardless of the CBA method used. The OCBAM method generally outperformed the OCBA method when used in the SIGA. However, the EQU and PTV methods also resulted in similar errors. Among the 21 different cases in this study, there are nine times the PTVS gave the best results and six times each that the EQUS and OCBAMS gave the best results.

To make these comparisons more formally, post hoc pairwise F-tests are applied to the data for the eight methods using the Matyas function with $D = 0.01$. The post hoc pairwise F-tests give the p-value of statistical comparison tests, comparing each method with each of the methods that performed worse than it. These tests provide insight into the statistical significance of the ranking of the methods for this problem. A significance level of $\alpha = 5\%$ is used, and the p-values associated with these tests are shown in Table 3.9. Several of the calculated p-values are smaller than the machine epsilon, and these are denoted in the table as '$< \epsilon$'. Statistically, significant results are indicated in the boldfaced text. These pairwise statistical comparisons induce a partial ordering of the GAs for this problem. This ordering is illustrated in Figure 3.7. For this problem, it can be seen that OCBAME is statistically dominated by the other methods and that OCBAMS, EQUS, and OCBAS are not statistically dominated by any other method. It can also be seen that each of the SIGA methods dominates each of the EIGA methods for this problem.

The dominance of SIGA methods over the EIGA methods is significant and easy to understand as the SIGA methods allocate samples as needed by the selection procedure and improves the quality of the GA. The difference between different allocation methods is more problem oriented and difficult to understand. The OCBA and OCBAM methods, which may be the best static allocation methods to identify the best individuals from a set, are not necessarily the best methods to identify the ordering of a set. The sampling with the OCBA and OCBAM methods is mostly allocated to the individuals that are currently the best ones or close to the best ones, focusing a large number of samples on those individuals and neglecting to sample other inferior individuals. When integrating with the EIGA, these two methods may greatly over-sample very few individuals and under-sample other individuals that might have the potential to exhibit better means when more samples are allocated. Thus, integrating OCBA or OCBAM methods with the EIGA can lead to worse solutions than the EQU or PTV methods with EIGA. However, the adverse impact on over-sample current good individuals can be greatly reduced when integrating with SIGA, since the tournament has a small size and the goal is to identify the best individual, the sampling allocation calculated by OCBA and OCBAM methods is more evenly and reasonably distributed than when integrating with EIGA or the other allocation methods, and the statistical results are greatly improved. Also, the reason that the OCBAM exhibits better solutions than OCBA may be caused by the fact that the OCBAM uses a distance measure that involves both information from the good individuals and the inferior individuals.

Table 3.9: P-values of post hoc pairwise comparison of all GA methods for Matyas function with $D = 0.01$ ($< \epsilon$ indicates p-value less than machine epsilon, bold-face text indicates statistical significance with significance level $\alpha = 0.05$)

Method	OCBAMS	EQUS	OCBAS	PTVS	EQUE	PTVE	OCBAE
EQUS	2.75×10^{-1}						
OCBAS	6.66×10^{-2}	1.83×10^{-1}					
PTVS	$\mathbf{1.82 \times 10^{-2}}$	6.77×10^{-2}	2.78×10^{-1}				
EQUE	$\mathbf{3.52 \times 10^{-7}}$	$\mathbf{6.42 \times 10^{-6}}$	$\mathbf{2.69 \times 10^{-4}}$	$\mathbf{2.05 \times 10^{-3}}$			
PTVE	$\mathbf{2.58 \times 10^{-8}}$	$\mathbf{6.23 \times 10^{-7}}$	$\mathbf{3.97 \times 10^{-5}}$	$\mathbf{3.94 \times 10^{-4}}$	3.13×10^{-1}		
OCBAE	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\mathbf{8.65 \times 10^{-13}}$	$\mathbf{2.51 \times 10^{-11}}$	
OCBAME	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\boldsymbol{< \epsilon}$	$\mathbf{1.11 \times 10^{-4}}$

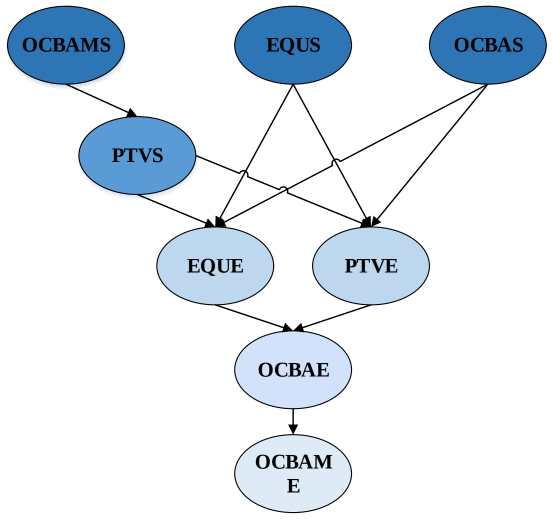

Figure 3.7: Partial ordering of all GA methods for Matyas function with $D = 0.01$ based on post hoc pairwise comparison with significance level $\alpha = 0.05$

Comparison of convergence

To visualize the effect that integrating the CBA methods into either the evaluation or the selection step of the GA has on the convergence of the algorithm, the OCBAMS, and the EQUE method are considered for the Matyas function with $D = 0.05$. These methods are generally the best SIGA and EIGA methods, respectively, performing the best for the greatest number of problems and noise levels. The errors of the best individual at each generation averaged over the 1600 trials are plotted in Figure 3.8. It can be seen that the errors associated with both algorithms initially decline at approximately the same rate. However, as the solutions approach the optimal solution, the OCBAMS method is better able to allocate the samples that it performs. This results in faster and continued progress during the later phase of the algorithm, resulting in a better solution. Alternatively, the OCBAMS method can reach a solution of the same quality as the EQUE method in approximately 200 generations compared with the 300 generations when using the same numbers of individuals and samples per individual per generation.

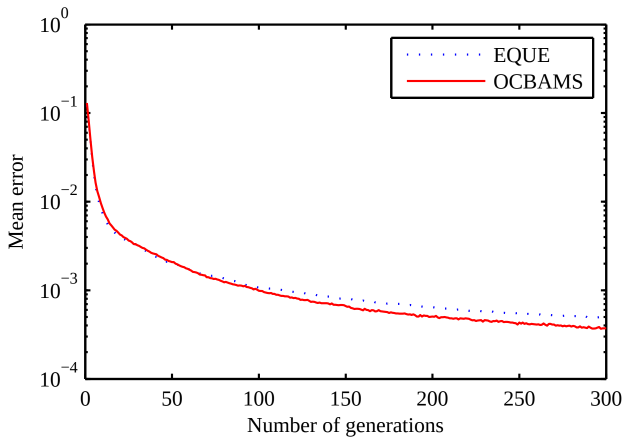

Figure 3.8: Example Convergence of EQUE and OCBAMS (Matyas function, $D = 0.05$)

Comparison of sampling efficiency

To understand the improvement of sampling efficiency due to the use of the SIGA, the EQU methods are selected as the baseline for the comparison since the improvement of other SIGA over EIGA are more significant than the EQU methods. The results of the EQUS for each problem and noise level are compared with the EQUE method with improved sampling. The number of allocated samples per individual n_1 for the EQUE method is increased in steps of 100 from 250 to 550, and the EQUE method was executed 1600 times to determine the mean error. A second order polynomial fit between the total number of samples per individual per generation n and the error of the EQUE method is established for each problem and noise level. This linear relationship is used to determine the equivalent number of samples per individual per generation required to obtain the same error with the EQUE method that was obtained with the EQUS method using 250 samples per individual per generation. An example of this process is shown in Figure 3.9. For each noise level ($D \in \{0.01, 0.05, 0.1\}$). The mean errors are plotted in the figure, and a linear relationship is extracted. Bars indicating the mean error obtained using the EQUS method with 250 samples and the equivalent number of samples as the EQUE method with same mean error are shown.

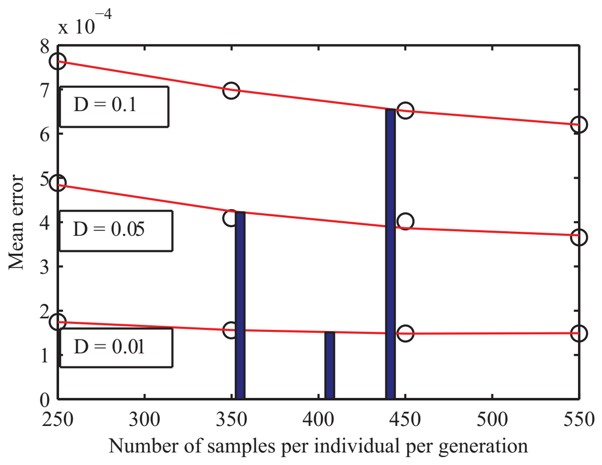

Figure 3.9: Equivalent numbers of samples per individual per generation for EQUE method to match the mean error of EQUS method with 250 samples per individual per generation for Matyas function (all three noise levels included) (circles denote the mean error by EQUE method with the corresponding number of samples per individual per generation, the lines indicate the second order polynomial relationship between number of samples per individual per generation and mean error, the bars indicate number of samples per individual per generation required by EQUE method to match mean error of EQUS method)

The calculated equivalent samples and the relative cost of obtaining the same error are shown in Table 3.10. It can be seen that the relative costs average approximately 143% with a maximal value of 225%. Generally, the relative cost of high noise levels is smaller than that of low noise levels. In such problems, the noise present in samples can dominate the differences in the expected fitness of individuals, requiring large numbers of samples for each individual. While there is still an improvement with using the SIGA in these cases, this improvement can be made more dramatic with a higher computing budget. Also, it can be seen in Table 4.2 that the SIGA results in a more dramatic improvement relative to the EIGA when other CBA methods are used.

Table 3.10: Equivalent numbers of samples per individual per generation and relative costs of EQUE versus EQUS to match EQUS mean error with 250 samples per individual per generation

Function	D	Equivalent Samples	Relative Cost
Matyas	0.01	407	163%
	0.05	355	142%
	0.1	441	176%
Rosenbrock	1	289	116%
	5	251	100%
	10	262	105%
Sphere	0.1	441	176%
	0.5	562	225%
	2	410	164%
Zakharov	0.1	511	204%
	1	468	187%
	2	356	142%
Griewank	50	343	137%
	250	260	104%
	500	263	105%
HappyCat	10	314	126%
	50	285	114%
	100	266	106%
HGBat	10	396	158%
	50	295	118%
	100	323	129%

3.4 Conclusion

An SIGA is proposed in which CBA methods are integrated into the selection process of the GA. This algorithm is compared with the EIGA, the typical existing approach of integrating CBA methods within the evaluation and ranking process of the GA. These algorithms are compared on several stochastic test problems with various levels of noise. It should be noted that only test functions with additive Gaussian noise are considered herein. This limitation is consistent with the traditional application of CBA methods [66]. However, future research may include the consideration of more general forms of stochastic problems. In particular, practical problems (e.g., design of shipboard power systems subject to hostile disruptions [67], a future application area) may have different forms. Furthermore, neither the existing EIGA nor the proposed SIGA can be directly applied to multi-objective stochastic problems, and the future extension to such problems would be of clear value.

In this study, it is found that the SIGA generally outperforms the EIGA regarding mean error for a given CBA method. This is attributed to the manner in which the SIGA allocates fitness evaluations towards specific individuals for whom the GA requires more information. It is also found that EIGA generally performs best with the EQU method of CBA. The performance of the SIGA is found to be less sensitive to the choice of CBA method. Finally, it is found that the SIGA can find solutions with comparable mean error to solutions found with EIGA when using the EQU method while requiring significantly fewer samples for the test problems. The average relative cost of the EQUE methods is 143% of that of the EQUS method and can be as high as 225%, but these costs vary with test function, noise level, and choice of CBA method.

CHAPTER 4
MULTI-OBJECTIVE GENETIC ALGORITHM WITH ALLOCATION TECHNIQUES FOR STOCHASTIC PROBLEMS

Real-world optimization problems are often multi-objective and stochastic problems [68,69]. Multi-objective evolutionary algorithms (MOEAs) are widely applied to solve those problems [70]. MOEAs combine operators such as mating selection [23], which selects child genes, crossover, and mutation to construct new generations of individuals. Among MOEAs, popular elitist approaches archive non-dominated solutions from the previous generation and combine them with non-dominated solutions from the current generation to produce the subsequent generation, a process which is referred to as environmental selection [23].

Researchers have developed different noise handling techniques to solve stochastic problems, aiming to improve the accuracy and efficiency of the algorithms. For example, a probabilistic method to improve sampling using loopy belief propagation for probabilistic model building genetic programming is described in [71]. Population statistics based re-sampling technique is introduced in [52] with a particle swarm optimization algorithm to solve stochastic optimization problems. For the elitist MOEAs, because of the stochastic nature of the objective functions, MOEAs must perform repeated computationally expensive samples to assess the fitness of each individual. The noise handling techniques seek to obtain more accurate results with fewer total evaluations. For example, the optimal computing budget allocation (OCBA) method proposed in [72] is integrated into the evaluation procedure to reduce the computing cost in [3]. In [73], fitness inheritance from parent genes is proposed to reduce the computational intensity required for evaluation. A probabilistic method based on statistical analysis of dominance is used to estimate Pareto-optimal front in [74].

In [75], confidence-based dynamic re-sampling is proposed to improve the confidence of Pareto ranking. A noise-aware dominance operator is integrated into the mating selection in [76]. However, due to the nature of the elitist MOEAs, environmental selection plays a more critical role than mating selection because it controls the evolving set of non-dominated solutions [77].

Herein, a fundamental question regarding the application of computing budget allocation (CBA) methods to MOEAs is considered. In particular, the effect of integrating CBA methods in either the evaluation or environmental selection on the accuracy of the MOEA is examined. In previous work with re-sampling applied as a noise handling technique, it is proposed either in evaluation [3] or in environmental selection [75]. However, there is no clear comparison between these two techniques; the comparisons focused only on whether the proposals improved the results or not. There exists no comprehensive study that examines, for a fixed total computing budget, where the re-sampling procedure should be integrated to improve the algorithms best. In this work, a combination of studies that compare the alternative approaches to integrating CBA methods into genetic algorithms (GAs), namely evaluation-integrated GAs (EIGAs) and selection-integrated GAs (SIGAs), are described. Various CBA techniques are compared, including the most basic equal allocation (EQU) method, OCBA method [72, 78], and the proportional-to-variance (PTV) method. These algorithms and CBA techniques are applied to stochastic multi-objective problems constructed from benchmark multi-objective optimization problems [79–81]. Numerical experiments are performed, and statistical testing is used to validate the significance of the comparisons.

The remainder of this chapter is organized as follows. First, descriptions of the considered stochastic multi-objective problems and different CBA methods are given in Section 4.1. In Section 4.2, the structure of the EIGA and SIGA approaches for integrating CBA methods into GAs are described. Test functions, performance metrics, and experimental results are presented in Section 4.3.

4.1 Multi-Objective Stochastic Problems and Computing Budget Allocation Methods

The multi-objective stochastic problems and the CBA methods considered herein are described below.

4.1.1 Stochastic problem statement

The multi-objective stochastic problems considered herein can be defined as

$$\min_{X} \; J_1(X), J_2(X), \ldots, J_H(X), \tag{4.1}$$

where X is the (possibly multi-dimensional) decision variable, J_1, J_2, …, and J_H are the H objectives to be minimized, $J_l(X) = E[L_l(X, \xi)], l \in \{1, 2, \ldots, H\}$, $L_l(\cdot, \cdot)$ is the sample performance of lth objective, and ξ is a random variable describing the problem noise. For the test problems discussed herein, the noise for each objective is modeled with additive independent and identical Gaussian distributions; however, it can also be represented in other ways. It is also assumed that such a problem is unconstrained in the sense that any constraints that bind the solution are appropriately penalized in $L(\cdot, \cdot)$.

For a deterministic multi-objective problem where $J_l(X) = L_l(X)$, the Pareto optimal front [82] is the complete set of non-dominated solutions. A solution for the multi-objective problem is defined as a non-dominated solution if it is not dominated by any other solutions. A solution a dominates solution b if $J_l(a) \leq J_l(b)$ $\forall l \in \{1, 2, \ldots, H\}$ and $\exists l \in \{1, 2, \ldots, H\}$ such that $J_l(a) < J_l(b)$. For the non-dominated solutions, each objective is minimized to the extent that it is not possible to further minimize one objective without making one or more other objectives bigger (worse).

For a stochastic multi-objective problem, the fitness function of each objective can only be estimated by calculating the mean of a limited number of random samples. For practical problems, it is assumed that the evaluation of the samples takes far more computation time and effort than the algorithm itself. Thus, it is desired to be able to allocate the samples, or the total computing budget, efficiently to obtain the best approximation of the Pareto optimal front. The quality of a Pareto optimal front approximation is measured by both its proximity to the true front and the degree to which it covers the true front.

4.1.2 Computing budget allocation methods

Various CBA methods have been studied, and the three such CBA methods applied in this study are discussed below. For each of these methods, it is assumed that N samples are being allocated among k individuals.

Equal allocation method

The simplest allocation technique to conduct sampling is the EQU technique, and it often serves as a benchmark for comparison [34]. The available computing budget is equally distributed to all individuals:

$$\tilde{n}_i = \frac{N}{k}, \tag{4.2}$$

where $\tilde{n}_i$ is the number of additional samples to be allocated to individual i.

Optimal computing budget allocation method

The OCBA method [72] for multi-objective optimization is based on maximizing the asymptotic probability that the selected subset is the non-dominated set. One such implementation is described below.

For a set of unique individuals S, S_P is defined as the non-dominated set and S_D is defined as the dominated set. In deterministic problems, S_P and S_D can be determined by non-dominated sorting [18]. The OCBA allocation rule aims to maximize the probability of correctly selecting the Pareto optimal set in stochastic problems.

For an individual i, which has previously been sampled, $\bar{L}_{il}$ is the sample mean, and σ_{il}^2 denotes the sample variance corresponding to the lth objective. For two individuals, i and j, the difference of sample means for objective l is expressed as

$$\delta_{ijl} = \bar{L}_{jl} - \bar{L}_{il}. \tag{4.3}$$

The individual that dominates i with the highest probability is approximated as

$$j_i \approx \arg \max_{j \in S, j \neq i} \prod_{l=1}^{H} P(L_{jl} \leq L_{il}) \approx \arg \min_{j \in S, j \neq i} \frac{\delta_{ijl_j^i} \left| \delta_{ijl_j^i} \right|}{\sigma_{il_j^i}^2 + \sigma_{jl_j^i}^2}, \tag{4.4}$$

where l_j^i denotes the objective for which j is better than i with the lowest probability and can be calculated as

$$l_j^i \equiv \arg \min_{l \in \{1, \ldots, H\}} P(L_{jl} \leq L_{il}) \approx \arg \max_{l \in \{1, \ldots, H\}} \frac{\delta_{ijl} \left| \delta_{ijl} \right|}{\sigma_{il}^2 + \sigma_{jl}^2}. \tag{4.5}$$

The set of individuals S is partitioned into subsets S_A or S_B based on the following equation:

$$S_A = \left\{ h | h \in S, \frac{\delta^2_{hj_h l^h_{j_h}}}{\sigma^2_{hl^h_{j_h}} + \sigma^2_{j_h l^h_{j_h}}} \leq \min_{i \in \Theta_h} \frac{\delta^2_{ihl^i_h}}{\sigma^2_{il^i_h} + \sigma^2_{hl^i_h}} \right\} \tag{4.6}$$

$$S_B = S \backslash S_A, \tag{4.7}$$

where

$$\Theta_h = \{ i | i \in S, j_i = h \}. \tag{4.8}$$

The samples are allocated to each individual based on its membership in S_A or S_B. In particular, for $h, m \in S_A$,

$$\frac{\tilde{n}_h}{\tilde{n}_m} = \left(\frac{\sigma_{hl^h_{j_h}} / \delta_{hj_h l^h_{j_h}}}{\sigma_{ml^m_{j_m}} / \delta_{mj_m l^m_{j_m}}} \right)^2. \tag{4.9}$$

For $d \in S_B$,

$$\tilde{n}^2_d = \sum_{h \in \Theta^*_d} \frac{\sigma^2_{dl^h_d}}{\sigma^2_{hl^h_d}} \tilde{n}^2_h, \tag{4.10}$$

where

$$\Theta^*_d = \{ h | h \in S_A, j_h = d \}. \tag{4.11}$$

Proportional-to-variance method

The PTV method utilizes the variance information from the existing samples. This method allocates computing budget proportional to the summation of the variance over all the objectives:

$$\frac{\tilde{n}_i}{\tilde{n}_j} = \frac{\sum_{l=1}^{H} \sigma^2_{il}}{\sum_{l=1}^{H} \sigma^2_{jl}}. \tag{4.12}$$

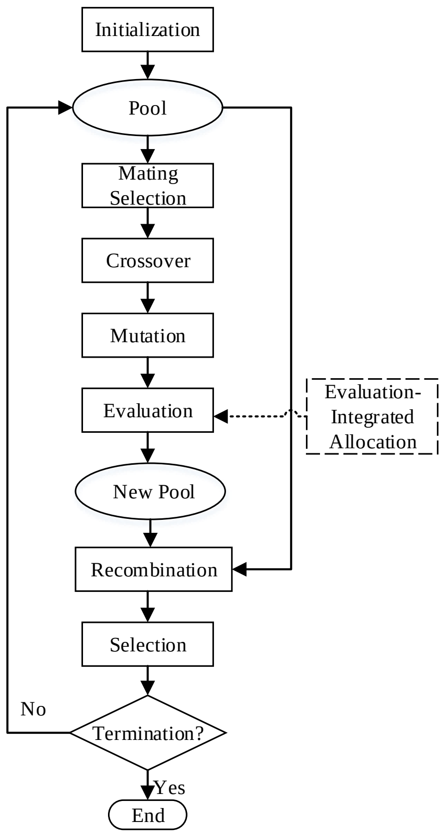

Figure 4.1: Flow chart of EIGA

4.2 Computing Budget Allocation Method Integrated Genetic Algorithms

NSGA-II is a widely used elitist GA for solving multi-objective optimization problems [18] . Herein, the CBA techniques described above are embedded in the framework of NSGA-II. Two methods of integrating CBA techniques into NSGA-II are considered. One is to integrate the allocation procedure into the evaluation step, and a GA using this approach is referred to as an EIGA [3]. The second is to integrate the allocation in the recombination and environmental selection procedure, which is to allocate the budget across the merged pool, and a GA using this approach is referred to as an SIGA. The flow charts of the integrated GA for either option are shown in Figure 4.1 and Figure 4.2.

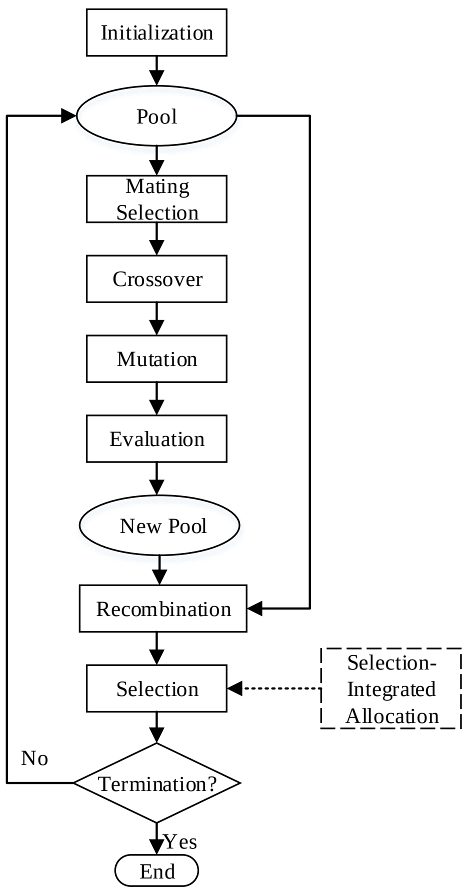

Figure 4.2: Flow chart of SIGA

Pseudocode for the two algorithms is listed in Algorithm 5 and Algorithm 6, and steps specific for each algorithm are marked in italics. The integrated GAs have the following steps:

Step 1: The population of n_{ind} individuals is uniformly randomly initialized in the search domain and is initially sampled n_0 times. An initial non-dominated sort is applied across the population [18]. The rank number and crowding distance [18] are assigned to each individual by the calculation of non-dominated sort in the fitness domain.

Step 2: Tournament selection is performed: for a total of n_{ind} randomly generated tournaments, each tournament contains two randomly selected individuals. Among

Algorithm 5 EIGA

```
1: Initialize population of individuals with random values in search space and evaluate the initial population with some initial samples; assign rank and crowding distance to each individual through non-dominated sort.
2: while Maximum generation not reached do
3:     Perform mating selection, selecting the individual out of each tournament with minimum rank and maximum crowding distance
4:     Perform crossover for each pair of individuals in the mating pool
5:     Perform mutation on each individual in the mating pool with selected mutation rate
6:     Evaluate each individual in the mating pool with some initial samples
7:     if EIGA then
8:         Apply CBA to allocate additional samples for each individual in the mating pool
9:     end if
10:    Combine the new pool of size $n_{ind}$ with the old pool of size $n_{ind}$, and generate a temporary pool of $2n_{ind}$
11:    Perform non-dominated sort on the temporary pool, and select $n_{ind}$ individuals to form the pool for the next generation based on rank and crowding distance
12: end while
```

Algorithm 6 SIGA

```
1: Initialize population of individuals with random values in search space and evaluate the initial population with some initial samples; assign rank and crowding distance to each individual through non-dominated sort.
2: while Maximum generation not reached do
3:     Perform mating selection, selecting the individual out of each tournament with minimum rank and maximum crowding distance
4:     Perform crossover for each pair of individuals in the mating pool
5:     Perform mutation on each individual in the mating pool with selected mutation rate
6:     Evaluate each individual in the mating pool with some initial samples
7:     Combine the new pool of size $n_{ind}$ with the old pool of size $n_{ind}$, and generate a temporary pool of $2n_{ind}$
8:     if SIGA then
9:         Apply CBA to allocate additional samples for each individual in the temporary pool
10:    end if
11:    Perform non-dominated sort on the temporary pool, and select $n_{ind}$ individuals to form the pool for the next generation based on rank and crowding distance
12: end while
```

each tournament, the individual with the smaller front number is selected. If the front numbers are equal, the individual with a larger crowding distance index is selected.

Step 3: Simulated binary crossover [17] and simulated binary mutation [18] are performed to generate the new pool of size n_{ind}.

Step 4: Each individual in the new pool is evaluated with n_0 number of samples. In the EIGA, the CBA method is used to allocate a total of $n_1 n_{ind}$ additional samples across the new pool.

Step 5: Combine the new pool and the old pool to form a temporary pool of size $2n_{ind}$. In the SIGA, the CBA method is used to allocate $n_1 n_{ind}$ samples to the temporary pool.

Step 6: Non-dominated Sorting is applied across the temporary pool to update the front index and crowding distance. The next generation pool is subsequently filled with n_{ind} individuals with minimum rank and maximum crowding distance.

Step 7: The algorithm terminates after n_{gen} generations. Otherwise, the algorithm continues at Step 2. The same number of total samples per generation are used in both EIGA and SIGA. The total number of samples allocated per algorithm is $(n_0 + n_1)n_{ind}n_{gen} + n_0 n_{ind}$.

The primary focus of this work is to compare EIGA with SIGA. NSGA-II is used as a representative multi-objective optimization framework, and the parameters for the GAs are given in Table 4.1. The parameters of the GAs are selected manually to achieve acceptable performance on the deterministic versions of the test functions included in this study. And they are used for both EIGA and SIGA. A different set of parameters can certainly affect the performance of the GAs, but since they are used as a baseline, the selection of these parameters is not the focus of this work. The sampling allocation procedure is performed sequentially, allocating the total samples in 10 steps, updating the sample means and variances between each step. The parameters are selected manually to achieve acceptable NSGA-II performance

Table 4.1: Parameters of Genetic Algorithms

Parameter	Parameter	Value
n_{gen}	Number of generations	500
n_{ind}	Number of individuals in pool	100
n_{tour}	Tournament size	2
η	Crossover constant	2
p_m	Mutation probability	0.05
n_0	Initial samples per individual	50
n_1	Allocated samples per individual	200

on the deterministic versions of the test functions described below and used for both the EIGA and the SIGA.

4.3 Results and Analysis

This section presents stochastic test problems and performance metrics, describes experimental results and compares the EIGA and SIGA as well as the CBA methods.

4.3.1 Test functions

Several multi-objective test problems are selected for constructing the stochastic problems. More specifically, ZDT1 and ZDT2 [79] are selected for examples of 2D test problems, and DTLZ2 and DTLZ6 [80,81] are selected for testing 3D cases. A typical setting of 30 variables in the ZDT test sets and 12 variables in the DTLZ test sets is followed. These problems are well-known test problems with known Pareto optimal sets. Independent zero-mean Gaussian noise is added to each objective of the deterministic test function:

$$L_l(X, \xi_l) = f_l(X) + \xi_l \tag{4.13}$$

$$\xi_l = \mathcal{N}(0, \sigma_l^2) \tag{4.14}$$

where σ_l^2 is the parameter establishing the noise level associated with the problem. In this study, values of σ_l of 1%, 10% and 20% of the maximum of each objective in the true Pareto-optimal set are used [83], indicating low, medium, and high noise levels, respectively. A letter 'L', 'M', or 'H' is appended to the names of the test problems to specify the specific stochastic test functions, e.g. 'ZDT1L' specifies the test function of ZDT1 with the low noise level.

4.3.2 Performance metrics

There are many different kinds of metrics measuring the performance of multi-objective optimization algorithms, such as general distance (GD) [84], maximum spread [85], hypervolume ratio [86], and inverse general distance (IGD) [87], and each metric may have different versions. Despite the diversity of metrics, they often measure two properties of the evolved front: how close is the evolved front to the true front and how well does the evolved front cover the true front. These two properties can be assessed with the GD and the IGD. The GD is calculated by averaging Euclidean distances from the evolved front to the true front and is the most widely used metric for convergence testing in multi-objective problems. The GD gives a good indication of the error between evolved front and the true front. The IGD is calculated by averaging the Euclidean distance from each true front sample to the evolved front and in this way, it conveys the measurement of both convergence and diversity. 1000 evenly distributed individuals on the true front are used to simulate the true front in the calculations. The GD is calculated as

$$GD = \sqrt{\frac{1}{n_{EPF}} \sum_{i=1}^{n_{EPF}} d_i^2}, \tag{4.15}$$

where n_{EPF} is the number of individuals in the evolved front, and d_i is the Euclidean distance of the individual i to the true front in the objective space, which is calculated by the distance of individual i to the closest among the 1000 evenly distributed individuals in the true front. The IGD is calculated as

$$IGD = \sqrt{\frac{1}{n_{TPF}} \sum_{i=1}^{n_{TPF}} \bar{d}_i^2}, \tag{4.16}$$

where n_{TPF} is the number of individuals in the true front, and $\bar{d}_i$ is the Euclidean distance of the individual i in the true front to the closest individual in the evolved front in the objective space. The GD and IGD metrics are two different metrics for the measurement of performance for multi-objective optimization, and they are intended to quantify the two properties of the evolved front. In both cases, lower values indicate better results, i.e., the evolved front is closer to the true front and well spread over the true front.

4.3.3 Comparison of different strategies

Experiments were performed to compare the EIGA and SIGA with different CBA methods under the various noise levels. The EIGA and SIGA with different CBA schemes are denoted as EQUE, EQUS, OCBAE, OCBAS, PTVE and PTVS with the last letter 'E' or 'S' denoting the EIGA or SIGA, respectively. Each algorithm was applied to each problem with each noise level using each CBA method 1200 times. Table 4.2 shows the mean GD and IGD for each method. The lowest value of GD and IGD for each problem and the noise level is indicated with boldfaced text. Graphical depiction of these results is shown in Figs. 4.3–4.6.

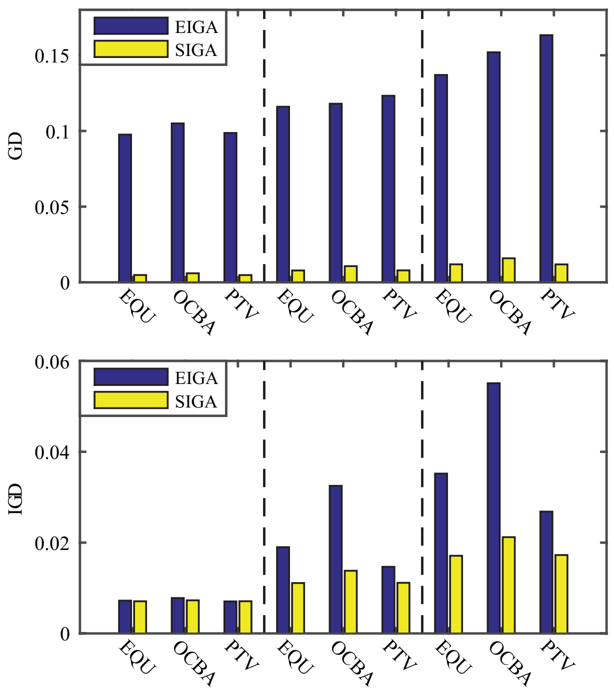

Figure 4.3: Comparison of general distance (GD) and inverse general distance (IGD) for ZDT1 function (the cases are divided by dashed lines from left to right corresponding to low, medium, and high noise levels)

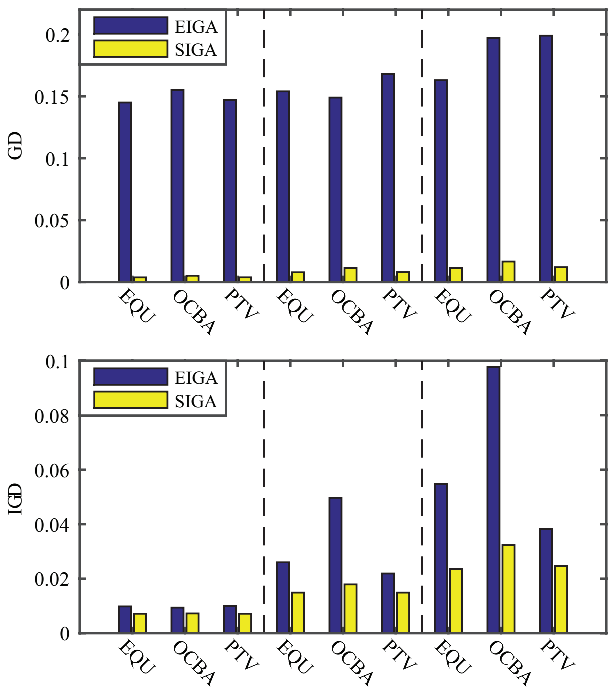

Figure 4.4: Comparison of general distance (GD) and inverse general distance (IGD) for ZDT2 function (the cases are divided by dashed lines from left to right corresponding to low, medium, and high noise levels)

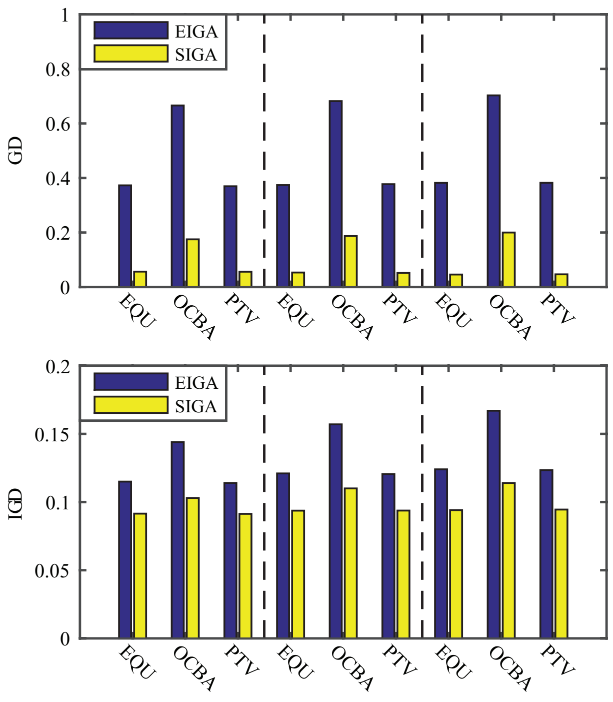

Figure 4.5: Comparison of general distance (GD) and inverse general distance (IGD) for DTLZ2 function (the cases are divided by dashed lines from left to right corresponding to low, medium, and high noise levels)

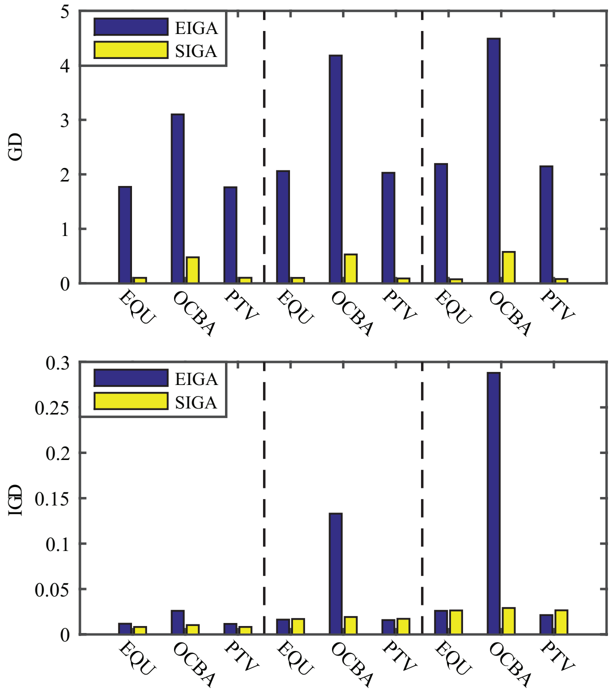

Figure 4.6: Comparison of general distance (GD) and inverse general distance (IGD) for DTLZ6 function (the cases are divided by dashed lines from left to right corresponding to low, medium, and high noise levels)

Table 4.2: Comparison of mean error for test functions (bold-face text indicates the best algorithm for each problem and noise level)

Function	Method	GD	IGD	Method	GD	IGD	Method	GD	IGD
ZDT1L	EQUE	9.76E-02	7.22E-03	OCBAE	1.05E-01	7.79E-03	PTVE	9.87E-02	7.05E-03
	EQUS	4.83E-03	**7.08E-03**	OCBAS	6.00E-03	7.29E-03	PTVS	**4.80E-03**	7.09E-03
ZDT1M	EQUE	1.16E-01	1.90E-02	OCBAE	1.18E-01	3.25E-02	PTVE	1.23E-01	1.47E-02
	EQUS	**7.85E-03**	**1.11E-02**	OCBAS	1.07E-02	1.38E-02	PTVS	7.91E-03	1.11E-02
ZDT1H	EQUE	1.37E-01	3.52E-02	OCBAE	1.52E-01	5.51E-02	PTVE	1.63E-01	2.68E-02
	EQUS	**1.19E-02**	**1.71E-02**	OCBAS	1.59E-02	2.12E-02	PTVS	1.19E-02	1.73E-02
ZDT2L	EQUE	1.45E-01	9.82E-03	OCBAE	1.55E-01	9.39E-03	PTVE	1.47E-01	9.94E-03
	EQUS	**3.79E-03**	**7.15E-03**	OCBAS	5.17E-03	7.24E-03	PTVS	3.82E-03	7.15E-03
ZDT2M	EQUE	1.54E-01	2.60E-02	OCBAE	1.49E-01	4.97E-02	PTVE	1.68E-01	2.19E-02
	EQUS	**7.89E-03**	**1.49E-02**	OCBAS	1.14E-02	1.79E-02	PTVS	8.00E-03	1.49E-02
ZDT2H	EQUE	1.63E-01	5.48E-02	OCBAE	1.97E-01	9.77E-02	PTVE	1.99E-01	3.82E-02
	EQUS	**1.15E-02**	**2.36E-02**	OCBAS	1.66E-02	3.23E-02	PTVS	1.20E-02	2.47E-02
DTLZ2L	EQUE	3.73E-01	1.15E-01	OCBAE	6.66E-01	1.44E-01	PTVE	3.70E-01	1.14E-01
	EQUS	5.66E-02	9.15E-02	OCBAS	1.75E-01	1.03E-01	PTVS	**5.63E-02**	**9.13E-02**
DTLZ2M	EQUE	3.74E-01	1.21E-01	OCBAE	6.82E-01	1.57E-01	PTVE	3.77E-01	1.21E-01
	EQUS	5.33E-02	**9.37E-02**	OCBAS	1.87E-01	1.10E-01	PTVS	**5.18E-02**	9.38E-02
DTLZ2H	EQUE	3.82E-01	1.24E-01	OCBAE	7.03E-01	1.67E-01	PTVE	3.82E-01	1.23E-01
	EQUS	**4.59E-02**	**9.41E-02**	OCBAS	2.00E-01	1.14E-01	PTVS	4.66E-02	9.45E-02
DTLZ6L	EQUE	1.77E+00	1.18E-02	OCBAE	3.10E+00	2.60E-02	PTVE	1.76E+00	1.16E-02
	EQUS	**1.01E-01**	**8.23E-03**	OCBAS	4.78E-01	1.03E-02	PTVS	1.02E-01	8.26E-03
DTLZ6M	EQUE	2.06E+00	1.63E-02	OCBAE	4.18E+00	1.33E-01	PTVE	2.03E+00	**1.58E-02**
	EQUS	1.00E-01	1.70E-02	OCBAS	5.30E-01	1.92E-02	PTVS	**8.99E-02**	1.72E-02
DTLZ6H	EQUE	2.19E+00	**2.60E-02**	OCBAE	4.49E+00	2.88E-01	PTVE	2.15E+00	2.13E-02
	EQUS	**7.50E-02**	2.64E-02	OCBAS	5.77E-01	2.91E-02	PTVS	7.96E-02	2.66E-02

Comparison of EIGA and SIGA

By examining Table 4.2 vertically, it can be seen that the SIGA with a given CBA method generally outperforms the EIGA with the same CBA method. The distances are assumed to follow a Rayleigh distribution, so an F-test can be used to determine the statistical significance of the results [65]. The null hypothesis is that the mean distance of the SIGA for a given CBA method is greater than or equal to the mean distance of the EIGA for the same CBA method. A significance level of $\alpha = 5\%$ is used to determine whether this hypothesis can be rejected. The results of these statistical tests are shown in Table 4.3. Due to a large number of samples (1200), many of the calculated p-values are smaller than the machine epsilon, and these are denoted in the table as '$< \epsilon$'. Statistically, significant results are indicated with boldfaced text. It can be seen that with the few exceptions in the IGD, the SIGA statistically significantly outperforms the EIGA for a given CBA method. For the GD metric, the SIGA was significantly better than the EIGA in each case considered.

Generally, it can be concluded that by integrating the CBA method into the selection process of the SIGA, the search algorithm allocates samples as needed to make the comparisons between individuals that are required by the algorithm and more accurately select the correct individuals into the next generation.

Comparison of computing budget allocation methods

The comparisons among different CBA methods can be made by examining Table 4.2 horizontally. The smallest GD and IGD among each case are boldfaced. It is shown that EQU and PTV allocation methods give similar results, and they are consistently better than the OCBA method. This may be due to the dynamic nature of GAs. It has been shown that the OCBA method works well in static selection problems [34], but the OCBA rule may not allocate samples as well as the population evolves.

Table 4.3: P-Value of Comparison between SIGA and EIGA for each CBA Method ($< \epsilon$ indicates p-value less than machine epsilon, boldfaced text indicates statistical significance with significance level $\alpha = 0.05$)

Function	EQU		OCBA		PTV	
	GD	IGD	GD	IGD	GD	IGD
ZDT1L	$< \boldsymbol{\epsilon}$	3.16E-01	$< \boldsymbol{\epsilon}$	5.21E-02	$< \boldsymbol{\epsilon}$	5.55E-01
ZDT1M	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	**3.24E-12**
ZDT1H	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$
ZDT2L	$< \boldsymbol{\epsilon}$	**4.44E-15**	$< \boldsymbol{\epsilon}$	**1.01E-10**	$< \boldsymbol{\epsilon}$	**4.44E-16**
ZDT2M	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$
ZDT2H	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$
DTLZ2L	$< \boldsymbol{\epsilon}$	**1.12E-08**	$< \boldsymbol{\epsilon}$	**1.11E-16**	$< \boldsymbol{\epsilon}$	**2.77E-08**
DTLZ2M	$< \boldsymbol{\epsilon}$	**2.00E-10**	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	**2.36E-10**
DTLZ2H	$< \boldsymbol{\epsilon}$	**7.53E-12**	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	**5.72E-11**
DTLZ6L	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$
DTLZ6M	$< \boldsymbol{\epsilon}$	8.48E-01	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	9.81E-01
DTLZ6H	$< \boldsymbol{\epsilon}$	6.46E-01	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	$< \boldsymbol{\epsilon}$	1.00E+00

From the previous discussion, the differences between SIGA and EIGA are dramatic and mostly statistically significant when using the same allocation method. However, for the same integration method, the differences between different allocation methods are less dramatic and may not be statistically significant, especially in the IGD metric. An example analysis is performed on ZDT1L. A post hoc pairwise F-test is applied to the data for the six methods, and a significance level of $\alpha = 5\%$ is used. The p-values associated with these tests are shown in Tables 4.4 and 4.5 for the GD and IGD, respectively. Several of the calculated p-values are smaller than the machine epsilon, and these are denoted in the table as '$< \epsilon$'. Statistically, significant results are indicated in the boldfaced text. These pairwise statistical comparisons induce a partial ordering of the GAs for this problem. In this specific example, the differences between the methods for the GD metric are mostly statistically significant, but for IGD metric the differences are largely not significant. The cases of ZDT1L,

Table 4.4: P-Values of Post Hoc Pairwise Comparison of Generalized Distance for all Methods on ZDT1L ($< \epsilon$ indicates p-value less than machine epsilon, boldfaced text indicates statistical significance with significance level $\alpha = 0.05$)

Method	PTVS	EQUS	OCBAS	EQUE	PTVE
EQUS	4.39E-01				
OCBAS	**2.39E-08**	**5.56E-08**			
EQUE	$< \epsilon$	$< \epsilon$	$< \epsilon$		
PTVE	$< \epsilon$	$< \epsilon$	$< \epsilon$	3.92E-01	
OCBAE	$< \epsilon$	$< \epsilon$	$< \epsilon$	**3.67E-02**	6.48E-02

Table 4.5: P-Values of Post Hoc Pairwise Comparison of Inverse Generalized Distance for all Methods on ZDT1L ($< \epsilon$ indicates p-value less than machine epsilon, boldfaced text indicates statistical significance with significance level $\alpha = 0.05$)

Method	PTVE	EQUS	PTVS	EQUE	OCBAS
EQUS	4.59E-01				
PTVS	4.45E-01	4.86E-01			
EQUE	2.80E-01	3.16E-01	3.28E-01		
OCBAS	2.06E-01	2.37E-01	2.48E-01	4.07E-01	
OCBAE	**7.26E-03**	**9.64E-03**	**1.06E-02**	**3.14E-02**	5.21E-02

DTLZ6M, and DTLZ6H are the cases that the IGD metrics are very close to each other. In the other problems, the differences are much larger.

The differences between different computing budget allocation methods for the same integration method may also rely on the properties of the specific function. Although these differences are less dramatic, the EQU and PTV allocation methods are more favorable than the OCBA method when integrated into NSGA-II.

4.4 Conclusion

Several noise handling techniques for multi-objective stochastic problems are studied. Various re-sampling techniques for allocating a given computing budget in either the environmental selection step or the evaluation process within an NSGA-II implementation are compared. These algorithms are compared on both 2D and 3D stochastic test problems with various levels of noise. It is found that the SIGA generally outperforms the EIGA regarding both GD and IGD metrics. This is attributed to the manner in which the SIGA allocates fitness evaluations towards specific individuals for whom the GA requires more information when evolving the Pareto front. Among the CBA methods, even though the OCBA method has been found to be a better method to maximize the probability of correct selection when not integrating with MOEAs, it is found to perform less favorably than the EQU and PTV methods when integrated into an elitist MOEA.

This work can be applied to solve practical multi-objective stochastic problems. Future study in the areas of optimization of photovoltaic inverters subject to solar irradiance variation [88] and the design of shipboard power systems subject to hostile disruptions [67, 89] will be performed to evaluate the relative merits of the CBA integration approaches on more practical optimization problems.

CHAPTER 5
APPLICATION OF GENETIC ALGORITHMS IN POWER ELECTRONICS

Many power electronics problems involve the optimization of discrete event dynamics systems (DEDS) [11] [90]. This type of problem can only be optimized through simulation and searching [91] [92]. Thus the selection integrated genetic algorithm can be applied to solve the problem efficiently by building models and using simulation for the evaluation. One example of utilizing genetic algorithms in power electronics problems is the optimization of grid-connected photovoltaic inverter system subject to random noises.

Nowadays, Renewable energy sources are becoming more important due to the environmental concerns [93]. Especially the solar energy, which is ultra clean, natural and a sustainable source of energy, has a large potential market than the limited and environmental unfriendly fossil and nuclear fuels [94]. Photovoltaic (PV) Inverter serves as the bridge to transfer solar power into electricity, and grid-connected PV system is becoming a commonly recognized method of contributing clean power to the grid [95].

However, the input power of solar energy, which subjects to weather changes, is a random and stochastic input to the PV system [96]; and the utility grid also contains a range of noises depending on the customer side usage status and the various utility side control [97]. Thus, the optimization and tuning of grid-connected PV inverter system subject to those noises have become a significant problem in both research and industry.

Based on the conclusion drawn from the previous chapter, the selection integrated generic algorithm is a good approach to solve multi-objective optimization problems with noise more efficiently. Vice versa, With the same number of samples, the SIGA can achieve better accuracy in stochastic problems. Power electronics system models usually contain complicated models and will take a long time to simulate due to the excessive zero crossings. It is proposed herein to use the multi-objective SIGA to tune the PV inverter system with noise. The remainder of this chapter is organized as follows:

First, the PV inverter system will be introduced in Section 5.1, including the modeling of the inverter and the control methods used in the system. Next, Section 5.2 will focus on the modeling of the noises that can are introduced in the PV inverter system. More specifically, a model of solar irradiance, which has a direct impact on the input power of the inverter system, is extracted and modeled from a one-day irradiance observation data file. And the model of grid harmonic, which causes pollution to the inverter currents if not appropriately controlled, is extracted and modeled from experimental measurement. Finally, the optimization methods and results will be given in section 5.3. Experimental validation of the optimization results in hardware is also included in this section. A short conclusion will be given afterward.

5.1 PV inverter system description and Control schemes

A grid-connected PV inverter system is composed of a PV array, an inverter, and a filter system, and a transformer which is used to step up the voltage and isolate current to the grid. In this study, the material or detailed modeling of PV array is not a primary concern. It is assumed that maximum power point tracking has been achieved for the system. And the PV array output is modeled as a constant power load, where the power output to the inverter system can maximumly tracking the solar power into the array. The transformer serves the purpose of damping the inverter output to the grid and isolating the current. It makes sure that no DC is flowing in the system. The primary focuses of this section are the modeling and controlling of the three-phase inverter part, and are illustrated as follows.

5.1.1 Modeling of three-phase inverter system

A basic three-phase inverter consists of six switches, with two switches in a pair for single-phase inversion. The operation of the three pairs is coordinated such that each pair corresponds to each phase for the three-phase inverter system. The three phase pairs operate identically except for the phase shift in between according to the control signal generated by pulse width modulation. It is assumed that the switches are ideal such that there is no power loss during the switching process. In realization of an inverter with digital controllers, the switches are controlled and are switching at a fixed switching frequency. The switching frequency is often limited by the hardware limitations. Currently, the power switches are dominated by IGBT and MOSFET. The IGBT applications usually are low frequency (less than 20KHZ) and high voltage (up to greater than 1000V). The MOSFET applications are usually high frequency (greater than 200KHZ) and low-voltage (less than 250V). In this study, IGBT switches at a switching frequency of 10KHZ are chosen.

Modeling of zero-order hold

Uniformly sampled pulse width modulation (PWM) is used to generate switching reference signal for digital implementation [98]. The modulation signal is regularly sampled at the beginning of the switching period and stored in a shadow register for use during the period [99]. The switching instant of uniformly sample PWM uses this value to compare to the carrier signal and produces an effective delay which is called zero-order hold and should be modeled appropriately. The following equation calculates the inversion from the DC side voltage to the three phase inverter input voltage.

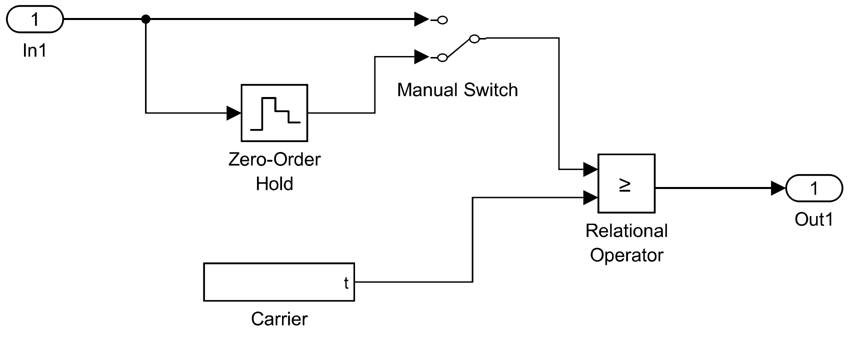

Figure 5.1: Block diagram of uniformly sampling PWM with zero-order hold

$$\mathbf{v}_{inv} = \frac{\mathbf{m}_{abc}+1}{2} H(ZeroOrder) v_{dc} - \frac{1}{3}\sum \mathbf{m}_{abc} v_{dc} \tag{5.1}$$

In which, the $\mathbf{v}_{inv}$ stands for the line to neutral voltage for the inverter input after the switching. The $\mathbf{m}_{abc}$ stands for the three-phase vector of the control signal produced by the circuit according to the controlling objectives. The v_{dc} is the DC input voltage before the switching. And the $H(ZeroOrder)$ part stands for the pulse width modulation with zero-order hold. A block diagram that depicts the uniform sampling PWM with zero-order hold is shown in Figure 5.1.

The transfer function of zero-order hold is as follows:

$$H(ZeroOrder) = \frac{1-\exp(-sT)}{sT} \tag{5.2}$$

T stands for the switching period, which is $1/f_{sw}$, f_{sw} stands for the switching frequency. The detailed modeling of the zero-order hold with the embedded SIMULINK block is very time consuming when running simulations due to its excessive zero-crossing. A second-order approximation of the zero-order hold is opposed to Eq. 5.2 using Taylor expansion series to improve the simulation speed of the model. Average model of the inverter system using the second order approximation can significantly improve the simulation speed without impairing too much accuracy.

$$H(ZeroOrder) \approxeq \frac{1}{\frac{T^2}{12}s^2 + \frac{T}{6}s + 1} \tag{5.3}$$

Modeling of LCL filter

A filter is required in the interconnection of the inverter to the grid. The purpose of a filter is to filter out the harmonics produced by the heavily switching. Basic filter topologies include L filter, LC filter and LCL filter [100]. The L filter is the first order filter with attenuation of 20dB per decade over the whole frequency range. Thus for high switching frequency applications, the attenuation is usually sufficient. However, it also greatly decreases dynamics of the whole system, and the damping may not be sufficient for lower switching frequency applications. The LC filter is second order filter and is better than L filter in damping behaviors [100]. However, there is a peak in resonant frequency that usually needs series or parallels damped. And there is a coupling issue on the filter and the grid impedance such that the grid impedance can affect the resonant frequency. The LCL filter is much better at decoupling the filter and the grid impedance than the LC filter. The current ripple over the grid inductor is also much lower. The attenuation of the LCL filter is 60dB per decade above resonant frequency and can be used for lower switching frequency cases. However, the LCL filter can be easily unstable, and there is a peak in cut-off frequency. A resistor damping is also suggested to improve the system stability and improve the behavior at cut-off frequency. Virtual resistor damping is also a good choice and is will be discussed next.

The LCL filter voltage and current calculations are governed by Eq. 5.4. The SIMULINK block which models the calculation of the voltage and current for the inverter side and the grid side is in Figure 5.2.

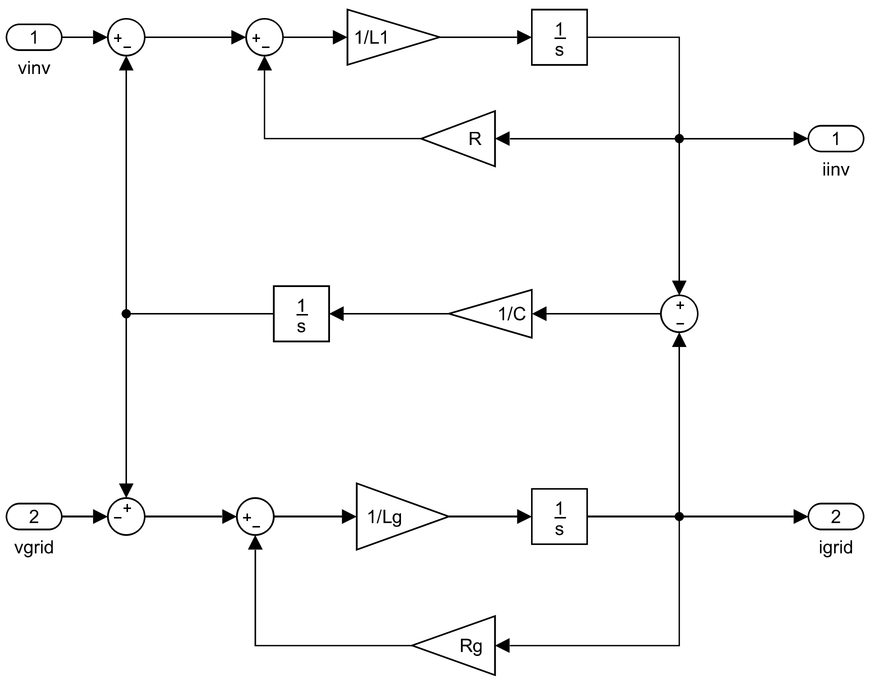

Figure 5.2: Block diagram of the LCL filter

$$\begin{aligned}\mathbf{v}_{inv} - \mathbf{v}_c &= L\frac{d\mathbf{i}_{inv}}{dt} + \mathbf{i}_{inv}R \\ \mathbf{v}_c - \mathbf{v}_{grid} &= L_g\frac{d\mathbf{i}_g}{dt} + \mathbf{i}_g R_g \\ \mathbf{i}_{inv} - \mathbf{i}_{grid} &= C\frac{d\mathbf{v}_c}{dt}\end{aligned} \tag{5.4}$$

The R and R_g are the equivalent series resistances (ESRs) associated with L and L_g of the LCL filter. The ESRs are measured because the inductors in real life are not ideal and they are usually measured with standardized frequencies. In the three-phase inverter application, the ESRs are most significant in the fundamental frequency, and they are approximated in Eq. 5.5

$$\begin{aligned}R &= 0.05 * \omega_0 * L \\ R_g &= 0.05 * \omega_0 * L_g\end{aligned} \tag{5.5}$$

where $\omega_0 = 2 * pi * 60$.

5.1.2 Control schemes for the three-phase inverter system

The two optimization objectives over the control of the three phase inverter are: 1) the input voltage of the inverter needs to be stabilized to the reference voltage with possible changes in the input solar power; 2) the current error of grid side current of the inverter needs to be minimized, this is in accordance to achieve less current harmonic and better control of current. The two objectives are obtained by a coupled proportional-integral (PI) control with a proportional resonance (PR) controlled inverter. PR controller is widely used in both single-phase and three-phase inverter systems due to its advantages over traditional PI control, as the PI control is known for several drawbacks: 1) presence of steady-state error in the stationary frame 2) the need to decouple phase dependency in three-phase system 3) possibility of distorting te line current caused by grid voltage harmonics [38]. In the study, LCL filter was used to filter out the high-frequency harmonics and active damping will be introduced.

The control diagram of the inverter system is shown in Figure 5.3. The control of the inverter takes the measurement of input DC voltage, the three-phase inverter side current, the three-phase grid side current and the three-phase grid voltage. The measurements are filtered with low pass filters to eliminate high frequency noises and the filtered measurements are denoted as v_{dcf}, $\mathbf{i}_{abcf}$, $\mathbf{i}_{abcgf}$, $\mathbf{v}_{abcgf}$ respectively. The reference provided to the control is the referenced input DC voltage, v_{dc}^{*}.

The goal stabilizing the input DC voltage is achieved by utilizing the PI controller to produce the reference current: the controller takes the voltage error from the DC voltage, then fed through the PI controller to produce the $i_{qe}^{\star}$ portion of the rotating reference frame. The $i_{de}^{\star}$ portion from the rotating reference frame is desired to be 0. The reference current vector $\mathbf{i}_{qde}^{\star}$ in rotating reference frame is then transformed to stationary reference frame $\mathbf{i}_{qd0}^{\star}$ for the inverter controller calculation. The reference

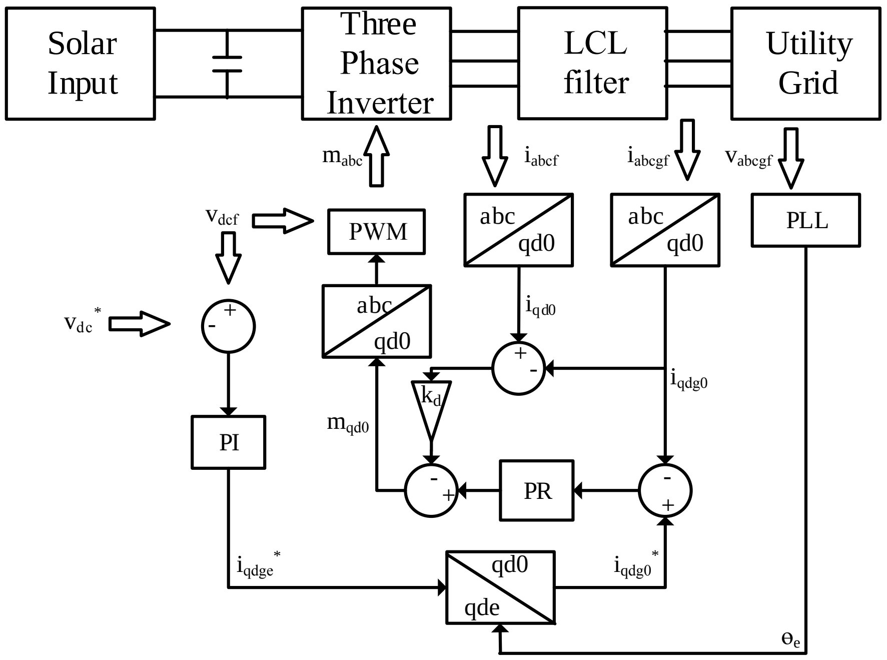

Figure 5.3: Control diagram of the complete PV inverter system

frame transformation from synchronous reference frame (abc) to synchronously rotating reference frame (qde) was introduced in section 2.4.2. The difference between rotating reference frame (qde) and stationary reference frame (qd0) is that the former takes instantaneous rotation angle θ into account and the later set the rotation angle as a constant of 0.

The PI controller $G_{pi}(s)$ for the reference current is defined in Eq. 5.6

$$G_{pi}(s) = K_{pi}(1 + \frac{1}{\tau_{pi}s}) \tag{5.6}$$

The reference current can be calculated following Eq. 5.7-5.8.

$$\mathbf{i}^{\star}_{qdge} = \begin{bmatrix} i^{\star}_{qge} \\ i^{\star}_{dge} \end{bmatrix} = \begin{bmatrix} G_{pi}(s)(v_{dcf} - v^{\star}_{dcf}) \\ 0 \end{bmatrix} \tag{5.7}$$

$$\mathbf{i}^{\star}_{qdg0} = \mathbf{K}_{e0}\mathbf{i}^{\star}_{qdge} \tag{5.8}$$

where $\mathbf{K}_{e0} = \mathbf{K}_r^{-1}\mathbf{K}_s$. The transformation coefficient of $\mathbf{K}_r$ and $\mathbf{K}_r^{-1}$ were given in Eq. 2.13 and Eq. 2.14. The transformation coefficient of $\mathbf{K}_s$ and $\mathbf{K}_s^{-1}$ can be obtained by plug in $\theta = 0$ into those equations. For the rotating reference frame, the rotating angle θ is calculated by performing PLL in section 2.4.2 on the grid voltage.

The PR controller is designed to control the grid current to reach the reference current. The filtered grid current $\mathbf{i}_{abcgf}$ is transformed into the stationary reference frame $\mathbf{i}_{qdg0}$ before calculation. The grid current usually contains 5th and 7th harmonic from the grid voltage distortion. To reduce the grid current distortion, the PR controller also includes harmonic compensator portion for 5th and 7th harmonic. The controller $G_{pr}(s)$ is defined in Eq. 5.9.

$$G_{pr}(s) = K_{pr}(1 + \frac{s}{\tau_{pr}(s^2 + \omega_0^2)} + \sum_{h=5,7} \frac{s}{\tau_{prh}(s^2 + (\omega_0 h)^2})\qquad(5.9)$$

The system for controlling the grid current using PR controller with LCL filter is easily unstable due to the second order system pole placement. The capacitor current feedback damping is used to move the pole placement of the overall system and increase the system stability. The modulation signal in the stationary reference frame m_{qd0} can be calculated from the output of PR controller and the capacitor current feedback damping part in Eq. 5.10.

$$\mathbf{m}_{qd0} = G_{pr}(s)(\mathbf{i}^{\star}_{qdg0} - \mathbf{i}_{qdg0}) - k_d(\mathbf{i}_{qd0} - \mathbf{i}_{qdg0})\qquad(5.10)$$

$\mathbf{m}_{abc}$ can be obtained by the transformation from the stationary reference frame to synchronous reference frame of $\mathbf{m}_{qd0}$. And PWM is used to generate switching reference signal based on the comparison of the modulation signal $\mathbf{m}_{abc}$ to a triangular switching signal.

The selection of the control parameters, including K_{pi}, τ_{pi}, K_{pr}, τ_{pr}, τ_{prh} and k_d, are a very difficult topic and there are papers on how to selecting reasonable parameters [38,101]. The remainder of this chapter will discuss different control cases that the PV inverter systems can face from the randomness from the solar input and the disturbances and harmonics from the utility grid, and how to use the multi-objective genetic algorithm with the integrated sampling strategies to select optimal control parameters for this stochastic problem.

5.2 Noise modeling from solar input and utility grid

As is discussed previously, the controlling objectives are to stabilize the input DC voltage under any circumstance of input power change caused by the irradiance change and to minimize the grid current error from the performance of the inverter and utility grid pollution. It is important to build models to represent the behaviors of the solar input and utility grid as accurate as possible to achieve the control goals.

5.2.1 Noise modeling of solar input

Modeling and forecasting of solar irradiance have been important research topic over the years [102,103]. Different numerical approaches are proposed to model the solar irradiance better. The use of a Markov chain-exponential model is used to generate precipitation of solar irradiance in [104].A Multilayer Perceptron MLP-model is proposed in [102] to forecast the solar irradiance on a base of 24h using the present values of the mean daily solar irradiance. A finite mixture of Dirichlet distributions is used to model the daily solar irradiance distributions in [105]. The models are trained to fit the practical data in these cases in a long term or a short term.

In this study, a set of one-day irradiance data from National Renewable Energy Laboratory taken in Oahu, Hawaii is collected, and models of mixture distribution by nonlinear regression are created to simulate the distribution of the irradiance and per

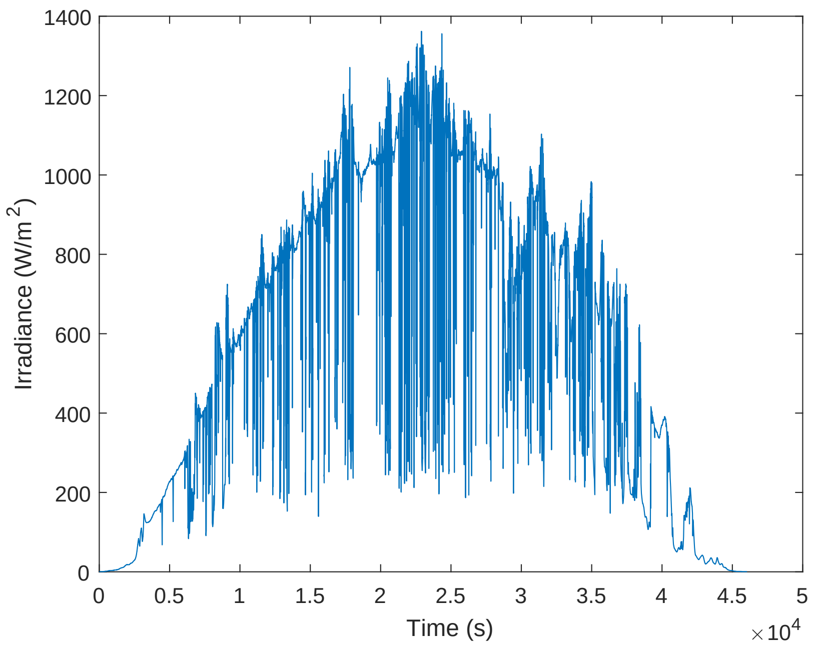

Figure 5.4: Figure of one day solar irradiance

second change of the irradiance. A plot of one-day solar irradiance from one measure point on the solar measurement grid is in Figure 5.4.

The mixture distribution fitting along with the histogram of the solar irradiance data is in Figure 5.5. The mixture distribution is modeled with a mix of Exponential distribution and T distribution. A representation of the probability density distribution (pdf) is in Eq. 5.11. Non-linear regression is performed to obtain the best-fit coefficients of the pdf on the real data.

$$\begin{aligned} f =& p(1) * exppdf(x * p(2), p(3)) \\ &+ p(4) * tpdf((x - p(5)) * p(6), p(7)) \\ &+ p(8) * tpdf((x - p(9)) * p(10), p(11)) \\ &+ p(12) * tpdf((x - p(13)) * p(14), p(15)) \end{aligned} \tag{5.11}$$

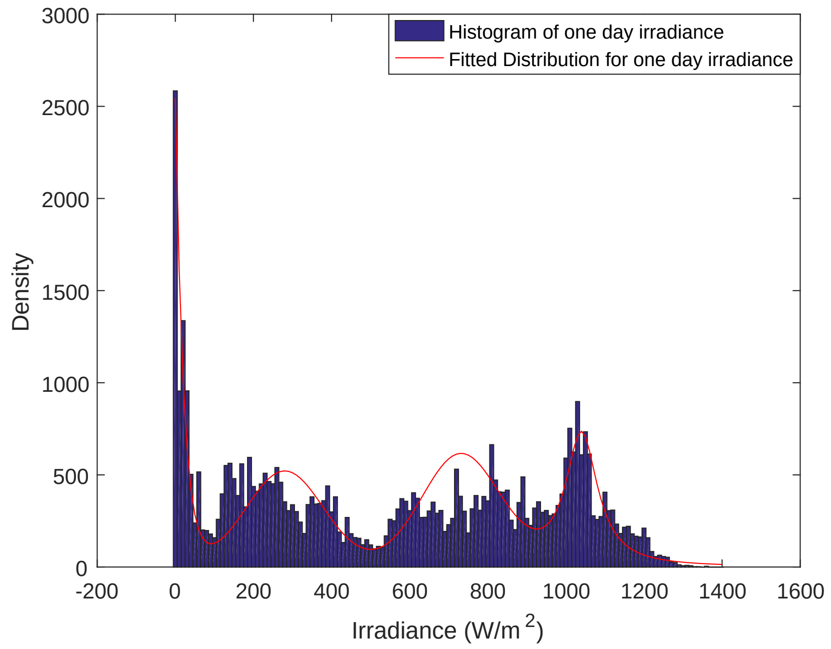

Figure 5.5: Histogram and distribution fitting of one day solar irradiance

where the trained coefficient is:

$$p = [3800\ 0.07\ 1.5\ 1300\ 280\ 0.01\ 500\ 1500\ 730\ 0.01\ 500\ 2300\ 1040\ 0.02\ 1] \tag{5.12}$$

To be able to model the per second change of the data. A mixture distribution with T Location Scale distribution and F distribution is fitted into the per second change of the irradiance shown in Figure 5.6. A representation of the probability density distribution is in Eq. 5.13. Non-linear regression is performed to obtain the best-fit coefficients of the pdf on the real data.

$$\begin{aligned} f =& p(1) * pdf(makedist(\text{`}tLocationScale\text{'}, \text{`}mu\text{'}, 0, \text{`}sigma\text{'}, abs(p(2)), \text{`}nu\text{'}, abs(p(3))), x) \\ &+ p(4) * fpdf((abs(x) - p(5))/(abs(p(6)) + 1), abs(p(7)), abs(p(8))) \end{aligned} \tag{5.13}$$

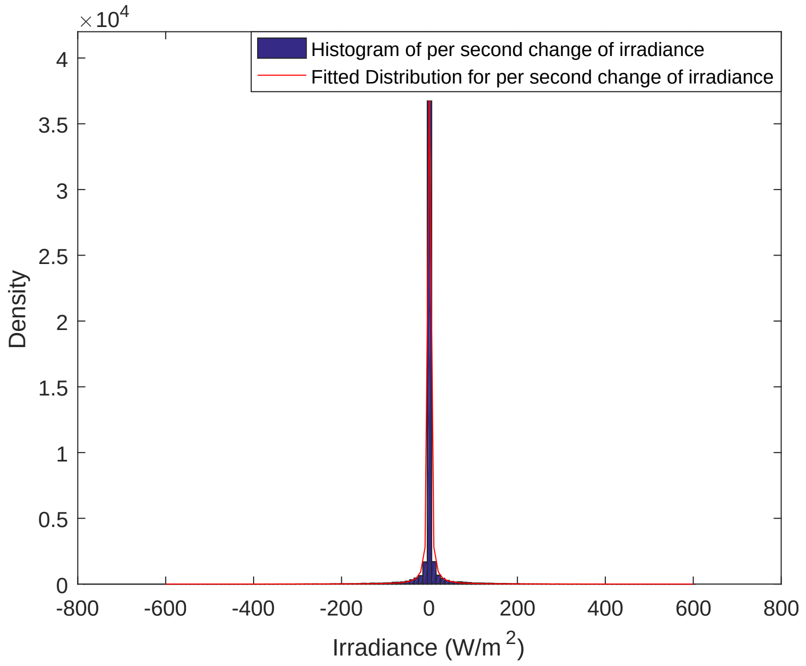

Figure 5.6: Histogram and distribution fitting of per second change of irradiance

where the best fit coefficient is:

$$p = [4.17e4\ 0.4527\ 3e7\ 2.4e4\ 1\ 3.6183\ 2.1169\ 1.9970] \tag{5.14}$$

After acquiring the probability density distribution (pdf) of the mixture distribution models, the solar irradiance input and per second change of solar irradiance in real time can be modeled as two random numbers that generated from the pdf. For the per second irradiance change distribution, note that the majority of the population are crowded in the middle. However, this distribution has long tails. To be able to sample the cases over the range of the distribution with a very limited number of samples, the stratified sampling technique is used to produce the random numbers in this case. The six stratas are [-600 -200], [-200 -100], [-100 0], [0 100], [100 200], [200 600]. Each of the strata is equally sampled. The stratified sampling ensures that the range of the distribution can be covered with a limited number of samples.

5.2.2 Modeling of utility grid

The utility grid is subject to grid harmonics [106]. And the grid is constantly changing with the plugging and unplugging of input injection and output load. Regulations and standards are posed to limit the grid variations [107,108]. Noises with the utility grid are modeled in this study to improve the performance of PR controller over the prominent harmonics. The grid voltage is modeled according to data acquiesced from a transformer which connects to the grid to the University of Kentucky during work days and holidays. The measurement is taken on the line to line voltage of v_{cb}. A Fourier transform analysis on the collected data suggests that among the harmonics, the 5th and 7th grid harmonic with phase angle shift are the most significant ones. During work days, there is more pollution over the fundamental frequency, while the grid has a cleaner spike on the fundamental frequency on holiday. A math model is constructed with the fundamental frequency, 5th and 7th harmonics with a phase shift. And non-linear regression with least squares error is performed on the math model and the acquired data. The equation for modeling the utility grid is in Eq. 5.15.

$$\begin{aligned} F =& 125 * (1 + x(1)) * \sqrt{2} * \sin(2 * pi * 60 * (t + x(2)) + pi/2) \\ &+ x(3) * \sqrt{2} * \sin(5 * (2 * pi * 60 * (t + x(2)) + pi/2 + x(5))) \\ &+ x(4) * \sqrt{2} * \sin(7 * (2 * pi * 60 * (t + x(2)) + pi/2 + x(6))) \end{aligned} \tag{5.15}$$

And the coefficients for the work day data and the holiday data are fitted in the least-squares sense, shown in Eq. 5.16

$$\begin{aligned} x_{workday} &= [-1.9972\ 0.0014\ 2.9684\ -1.8392\ 1.6983\ -2.6631] \\ x_{holiday} &= [0.0058\ 0.0005\ 2.3177\ -1.9644\ 2.2869\ -2.2230] \end{aligned} \tag{5.16}$$

Visualization of the fitting results along with the original data are presented in Figure 5.7 and Figure 5.8.

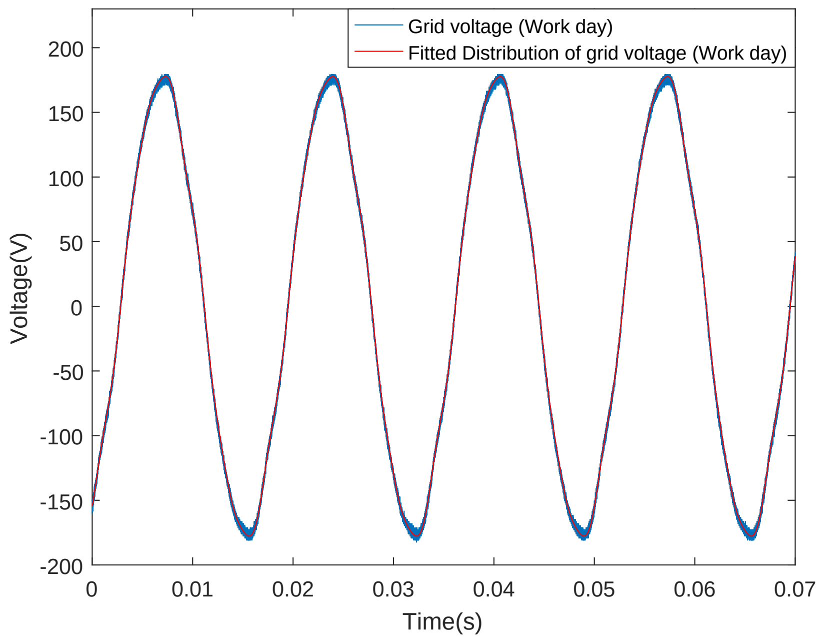

Figure 5.7: Measurement and distribution fitting of utility grid (Work day)

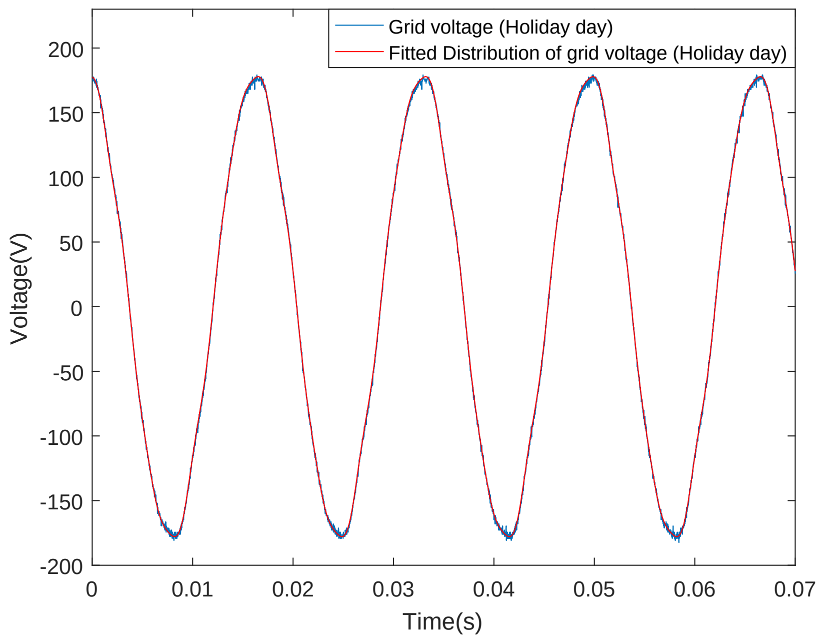

Figure 5.8: Measurement and distribution fitting of utility grid (Holiday day)

Based on the extracted models. Noises or errors about the utility grid are estimated with the assumption that the noise from the grid follows a White Gaussian distribution with the expected mean values. The pdf of the RMS fundamental voltage (V_{grid}), the RMS 5th harmonic voltage (V_{5th}), the RMS 7th harmonic voltage (V_{7th}), the phase shift of 5th harmonic (P_{5th}) and the phase shift of 7th harmonic(P_{7th}) are defined in Eq. 5.17.

$$\begin{aligned}
V_{grid} &= 125 * (1 + \mathcal{N}(0, 0.01)) \\
V_{5th} &= 2.5 * (1 + \mathcal{N}(0, 0.5)) \\
V_{7th} &= 2 * (1 + \mathcal{N}(0, 0.5)) \\
P_{5th} &= 2 * (1 + \mathcal{N}(0, 0.5)) \\
P_{7th} &= 0.7 * (1 + \mathcal{N}(0, 0.5))
\end{aligned} \tag{5.17}$$

5.3 Stochastic optimization and Hardware validation

Due to the stochastic nature of the solar power input and the output grid, the proposed control scheme may produce large voltage ripple, heavily distorted grid current or even instability if the control parameters are not well tuned for all the possible cases. From section 5.1, some guidance and potential range of the control parameters can be defined. However, it is impossible to manually select best solutions which might fit all the noisy cases. The multi-objective genetic algorithm with selection integrated scheme is proposed to tune the parameters and produce a Pareto front of the two control objective.

5.3.1 Optimization setup

A MATLAB simulation of the proposed inverter system with the modeling of power input and utility grid is built and parameters for the power input and utility grid can be adjusted according to the noise modeling. Due to the limitation of the hardware environment, the solar power input with randomness is represented with a DC power supply. The DC power supply can mimic the solar power by running under current limit mode with the desired voltage. The initial voltage of the power supply is set at 260V. And the initial current of the DC power supply is limited to a small amount for the initiation of the inverter. After turn on the inverter and the control of the inverter can adjust the input voltage to a steady DC voltage of 250V. The power knob of the DC power supply can be turned to increase or decrease to mimic the solar power change. The inverter will transfer the power from the DC power source to the utility grid. The modeling of the DC power source in current limitation mode is described in Eq. 5.18.

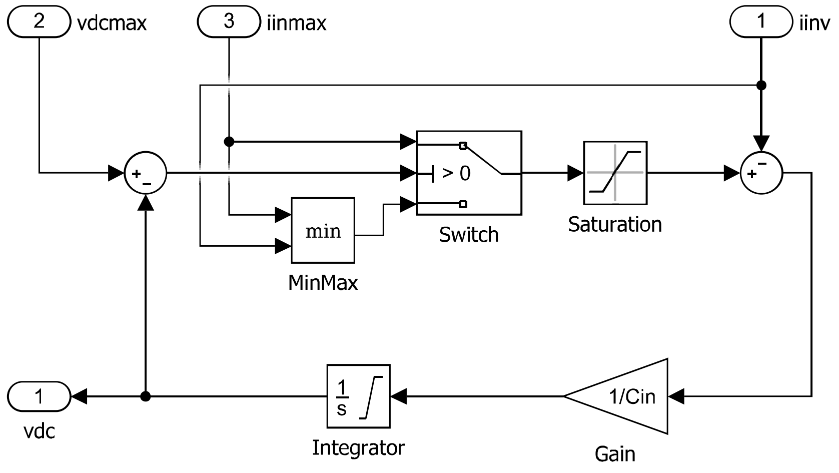

Figure 5.9: Model of DC power source connect to inverter

$$i_{in} = \begin{cases} i_{inmax}, & v_{dcmax} >= v_{dc} \\ \min(i_{inmax}, i_{inv}), & v_{dcmax} < v_{dc} \end{cases} \tag{5.18}$$

where the voltage from the power supply is denoted as v_{dcmax}; the inverter capacitor voltage is denoted as v_{dc}; the current from the power supply is denoted as i_{inmax}; the current flowing in the inverter is denoted as i_{inv}.

The SIMULINK model of the source is in Figure 5.9.

The randomness of the solar input and utility grid is introduced by generating random parameters for the simulation according to the distributions defined in the last section. The input voltage of the inverter is fixed at the desired DC voltage of 250. Thus the output current of the DC power supply is proportionally scaled to mimic solar power from the solar irradiance. The ramp change of the output current is also proportionally scaled to mimic the power change from per second solar irradiance change. The scaling needs to ensure that the maximum irradiance of the day, which is 1400 corresponds to a current which is within the maximum current limit of the hardware environment. A tolerable operation current is around 30A. The scaling relationship is described in Eq. 5.19. The current is defined to be no less

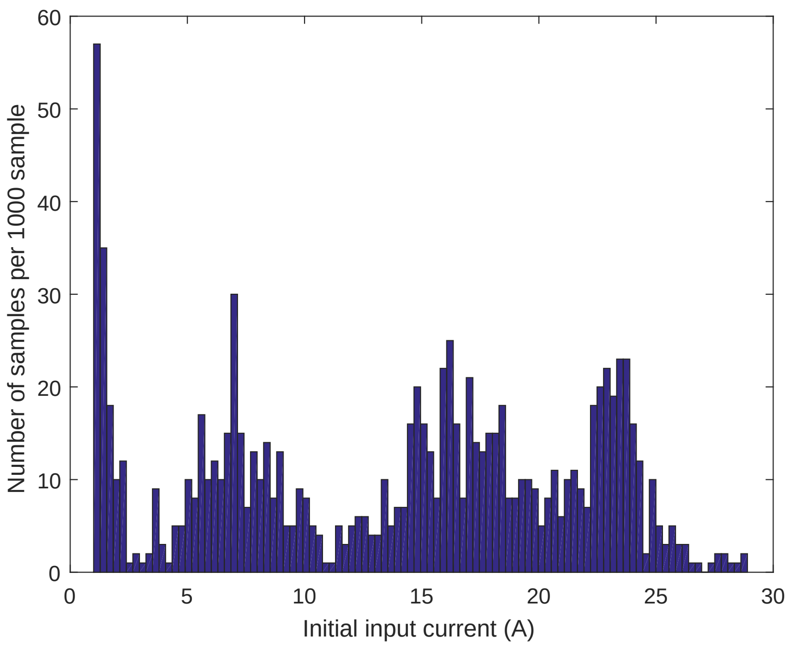

Figure 5.10: A distribution of generated input current scaled to the solar irrdiance input for 1000 sample

than 1A. The per second current change, which is the ramp current, is then restricted accordingly.

$$
\begin{aligned}
i_{initial} &= IR_{initial} * 30/1400 + 1 \\
i_{ramp} &= \begin{cases} IR_{ramp} * 30/1400 & IR_{ramp} >= -IR_{initial} \\ -IR_{initial} * 30/1400, & IR_{ramp} < -IR_{initial} \end{cases}
\end{aligned} \tag{5.19}
$$

where $i_{initial}$ stands for the initial output current from the power source, and i_{ramp} stands for the per second current change; $IR_{initial}$ stands for the initial irradiance generated from the irradiance distribution, and IR_{ramp} stands for the per second irradiance change generated from the per second irradiance change distribution.

The histograms of a set of random samples with the sample size of 1000 are shown in Figure 5.10-5.14.

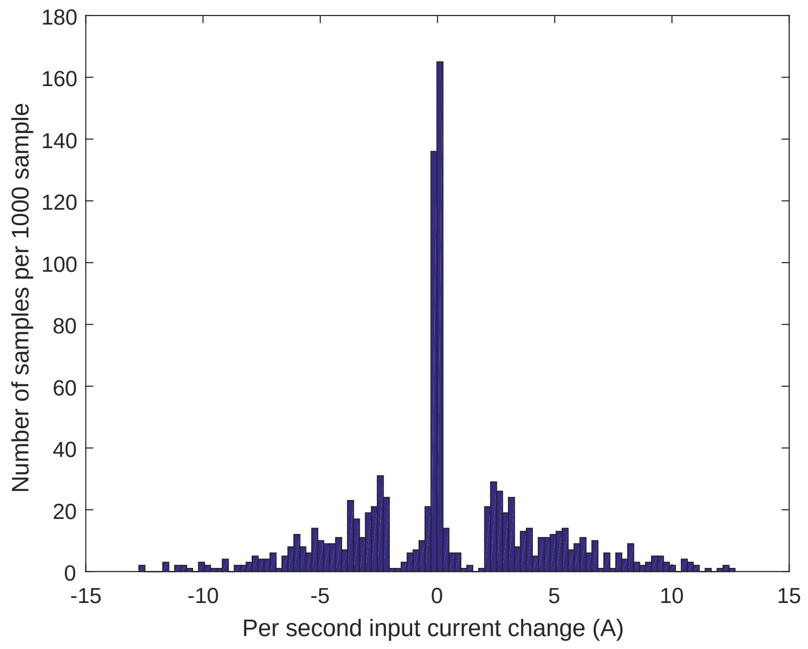

Figure 5.11: A distribution of generated per second current change scaled to the per second irradiance change for 1000 sample

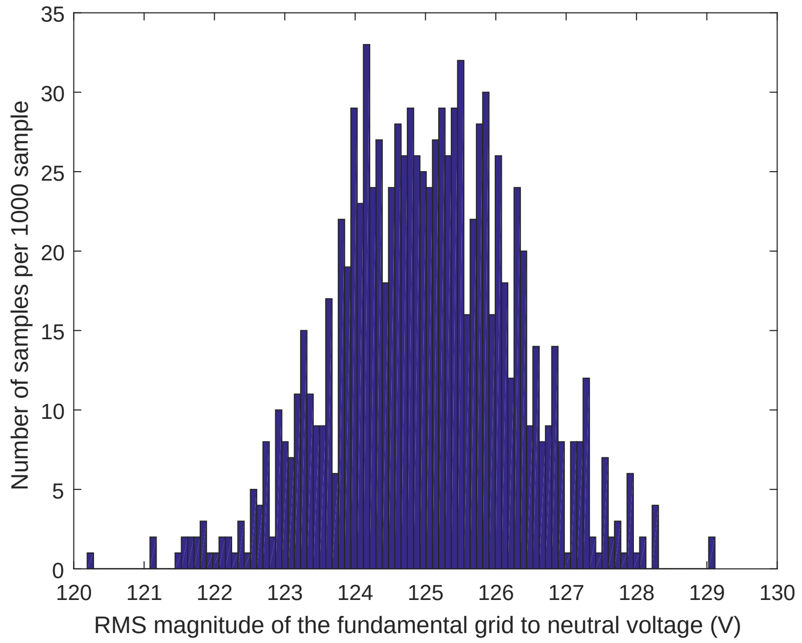

Figure 5.12: A distribution of generated RMS magnitude of the fundamental grid to neutral voltage for 1000 sample

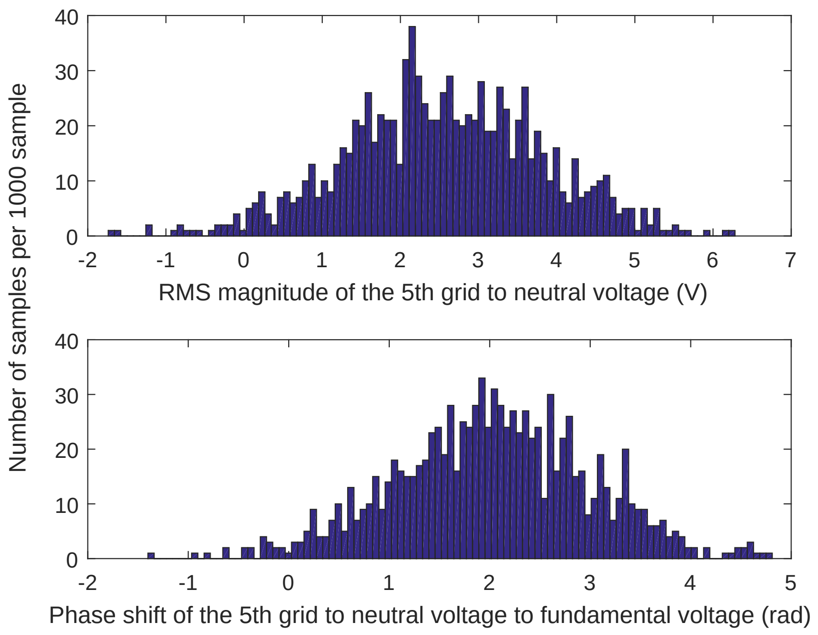

Figure 5.13: A distribution of generated RMS magnitude and phase shift of the 5th grid to neutral voltage for 1000 sample

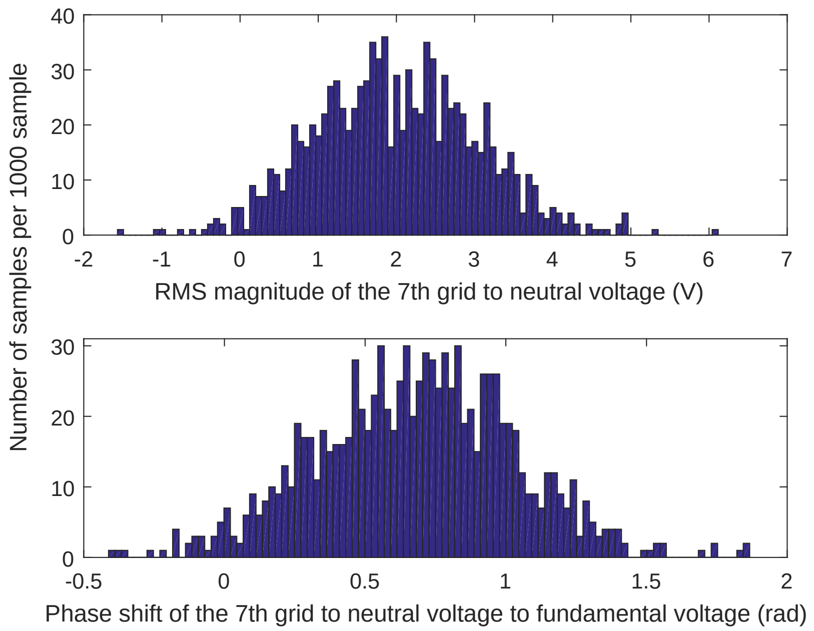

Figure 5.14: A distribution of generated RMS magnitude and phase shift of the 7th grid to neutral voltage for 1000 sample

Table 5.1: Parameters of Genetic Algorithm

Variable	Parameter	Value
ngen	Maximum Generation	50
nind	Initial Pool	100
npool	Mutation Pool	50
ntour	Tournament size	2
nsamp	Initial sample per individual	50
tsamp	Allocated sample per individual	50
eta	Crossover constant	2
pm	Mutation constant	0.05

The SIGA is used to tune the control parameters on the simulation model with varying solar input and the utility grid. The simulation is set such that the system will run 1 second on initial current for the system to reach steady state, then followed by 1 second of ramp current to simulate the per second irradiance change, then the system will continue in the new state for 0.5 seconds. Simulation results from 0.5 seconds to 2.5 seconds are recorded for fitness evaluation. The parameters of the GA are specified in Table 5.1. The ranges of the tuning control parameters are in Table 5.2. Other fixed filter parameters and control parameters are in Table 5.3. The other parameters including the three phase inverter hardware setup are in Table 5.4. The hardware structure of the three-phase inverter are shown in Figure 5.15-5.16.

The fitness values for the fitness function are calculated from the simulation output. The fitness function is defined as the voltage error and grid current error defined in Eq. 5.20. The grid current error is calculated in the rotating reference frame. This is because that in steady state, the grid current in the rotating reference frame are two constants, and the error can be easily calculated. However, in the case of the

Table 5.2: Searching Range for tunable control parameters

Variable	Parameter	Value
K_{pi}	Proportional gain for PI controller	[1e-2 1e2]
τ_{pi}	Integral constant for PI controller	[1e-4 1]
K_{pr}	Proportional gain for PR controller	[1 10]
τ_{pr}	Integral constant for PR controller	[1e-4 1]
τ_{pr5}	Integral constant for PR controller	[1e-4 1]
τ_{pr7}	Integral constant for PR controller	[1e-4 1]
k_d	Capacitor current feedback damping coefficient	[1 10]

Table 5.3: Un-tunable control and filter parameters

Variable	Parameter	Value
kp_{pll}	kp for phase locked loop	0.1
ki_{pll}	ki for phase locked loop	7.5398
τ_{DC}	filter time constant for DC voltage	$1/(2 * pi * 100)$
τ_f	filter time constant for currents	$1/(2 * pi * 2000)$

Table 5.4: Parameter for Three Phase Inverter

Variable	Description	Value
L	Inverter side inductor	0.276 mH
L_g	Grid side inductor	0.4 mH
R	Inverter side inductor ESR	5.2 $m\Omega$
R_g	Grid side inductor ESR	7.5 $m\Omega$
C_f	Filter capacitance	24 μF
C_{in}	Input capacitance	680 μF
v_{dc}^*	Input referenced voltage	250 V
f_{sw}	Switching frequency	10 kHz

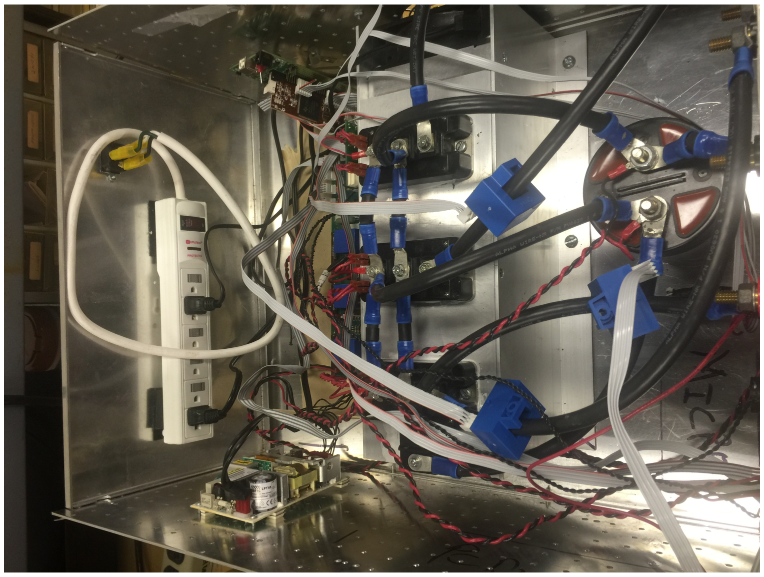

Figure 5.15: Hardware box for the switches, input capacitor and controller

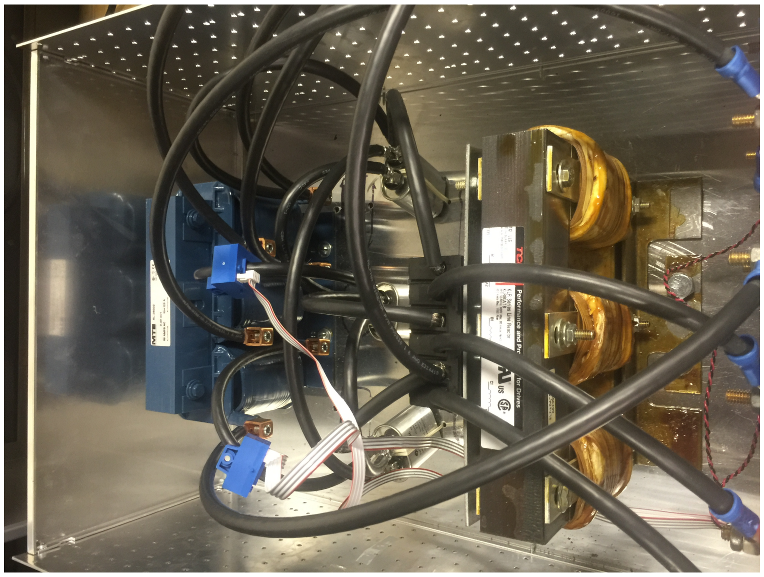

Figure 5.16: Hardware box for the LCL filter

simulation is unstable, the DC voltage and grid current will be out of normal range, and the simulation is terminated in advance when out of range is detected. The simulation ending time t_{end} would be smaller than the simulation final time t_{final}. The penalty for these cases are described in Eq. 5.20. The purpose of the penalty function are to distinguish the unstable cases from the stable cases, and also make sure more penalty would be applied when the system is more unstable, which takes less time to terminate. The goal of the SIGA is to minimize the voltage error and the current error. And with these settings, the optimization goal can be achieved, and results are discussed in next part.

$$F_{fitness} = \begin{cases} \begin{bmatrix} 1e9 * (10 - \frac{t_{end}}{t_{final}}) \\ 1e9 * (10 - \frac{t_{end}}{t_{final}}) \end{bmatrix} & t_{end} < t_{final} \\ \begin{bmatrix} \sqrt{\frac{1}{t_{final}} \int (v_{dc}^{\star} - v_{dc})^2 dt} \\ \sqrt{\frac{1}{t_{final}} \int (i_{qge}^{\star} - i_{qge})^2 + (i_{qge}^{\star} - i_{qge})^2 dt} \end{bmatrix} & t_{end} = t_{final} \end{cases} \tag{5.20}$$

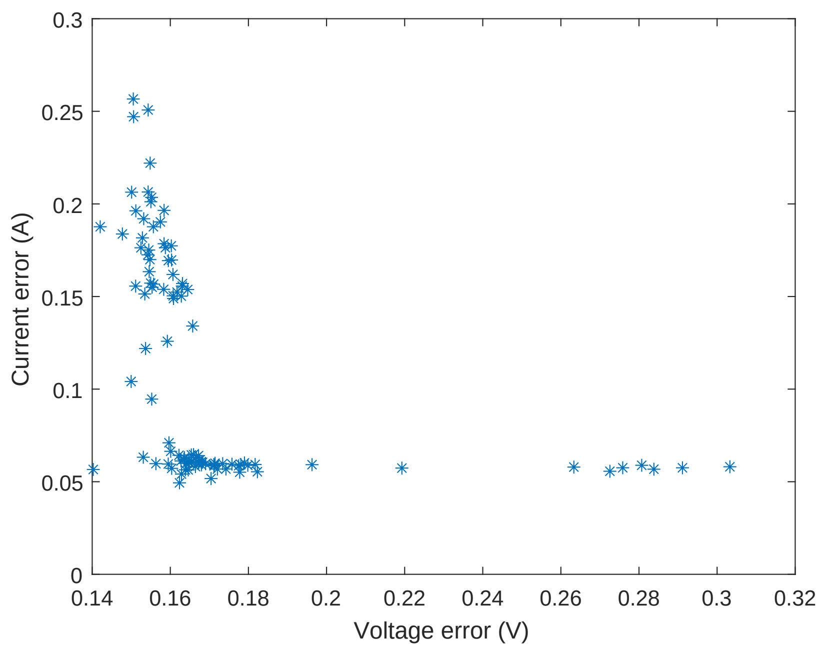

Figure 5.17: Last generation population from the SIGA

5.3.2 Results and validation

The results and validation for the optimization of the stochastic three-phase inverter system problem are described in this subsection. The last population of the individuals from the SIGA is given in Figure 5.17. Due to the limited number of samples during the SIGA, the fitness values are not good estimates of "true mean" values. To give a better interpretation of the "true mean" value for the last population, the fixed set of 1000 samples from Figure 5.10-5.14 is evaluated on the last populate and the fitness is given in Figure 5.18. It can be seen that despite the limited number of samples, the SIGA did converge very well to the Pareto front.

To better observe the trade-off between the voltage error and current error objectives. Selected cases of 'A', 'B' , 'E1', 'E2' and 'C' are listed in Table 5.5. It can be observed that the points from left to right in Figure 5.18 generally show an increasing trend in voltage error and a decreasing trend in current error.

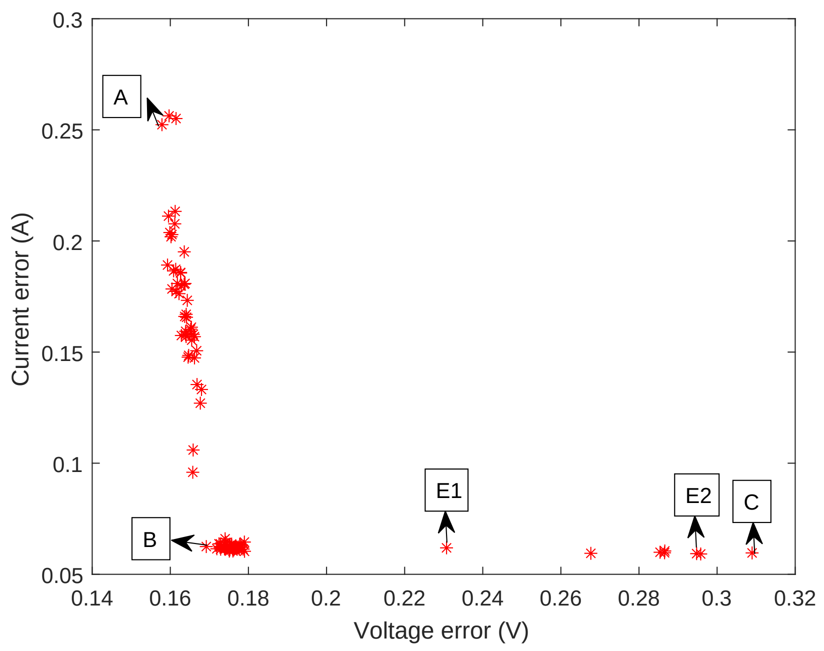

Figure 5.18: Last generation population from the SIGA with 1000 sample re-evaluation

Table 5.5: Comparison of Case A, B, E1, E2 and C

	Case A	Case B	case E1	case E2	Case C
Voltage error (V)	0.1579	0.1692	0.2307	0.2948	0.3090
Current error (A)	0.2523	0.0625	0.0619	0.0593	0.0596

The simulation comparison of grid current in rotation reference frame and DC side voltage of the end cases ‘A’ and ‘C’ under the same input is shown in Figure 5.19-5.20. The initial current is 5A and ramp current per second is 10A for the simulation. The grid values are the mean values of each parameter. Case ‘C’ has less current harmonic content than case ‘A’ with more stable rotation reference frame currents. A phase Figure 5.20 shows that case ‘A’ has more stable DC voltage than case ‘C’.

By comparing the three cases, the ‘knee’ point of the front, which is case ‘B’ is usually the most desired selection for a balance of the two objectives. Both current error and voltage error are regulated very well at this point. However, the PI control for case ‘B’ and its left points are heavily under damped such that during the initial

Table 5.6: Harmonics in steady state from experiment (Exp) and simulation (Sim)

Case	$\text{Exp}_{5\text{th}}$ (%)	$\text{Exp}_{7\text{th}}$ (%)	$\text{Sim}_{5\text{th}}$ (%)	$\text{Sim}_{7\text{th}}$ (%)
E1	0.7585	1.4054	0.0323	0.0403
E2	0.5935	1.3275	0.0307	0.0378

transient of starting the inverter, the DC voltage would drop below 250 significantly and power flow reversely from the grid to the power source, which is not allowed in the present experiment environment. Due to this concern, case 'E1' was selected to represent the lower voltage error and higher current error case in hardware experiment. And case 'E2' was selected to represent the higher voltage error and lower current error case since case 'E2' exhibits better performance in both objectives than case 'C'. The power source was initialized with 260 V DC output in current limit mode. Then the knob to change the current was turned clockwise to increase the input current which simulates the per second power change. Since each set of experiment is human performed and the initial current and change current should be different each time. So voltage error quantitatively calculated is unable to provide a fair comparison between the two cases and can only be qualitatively examined. During the two sets of experiments, very similar operations are performed and the initial current and final current are controlled very close to each other within 1 A deviation. The measured DC voltages for case 'E1' and case 'E2' are shown in Figure 5.21-5.22. The A phase currents for the two cases are shown in Figure 5.21-5.22. It can be observed that the transient for case 'E2' is larger than the transient for case 'E1'. Current data from last 0.5 second are used to calculate the 5th and 7th harmonic percentages shown in Table 5.6. A set of harmonic percentages from simulation is also included with the same simulation setting. It shows that case 'E2' exhibits larger 5th and 7th current harmonic than case 'E1', which indicates that case 'E2' exhibits larger current error in rotation reference frame. The experiment results correspond to the simulation results from Table 5.5.

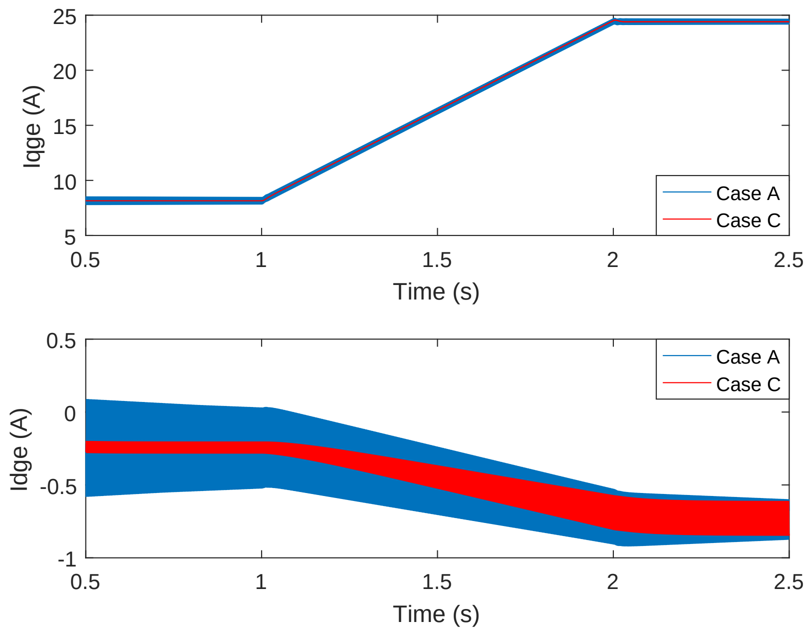

Figure 5.19: Comparison of current between case 'A' and case 'C' in rotation reference frame

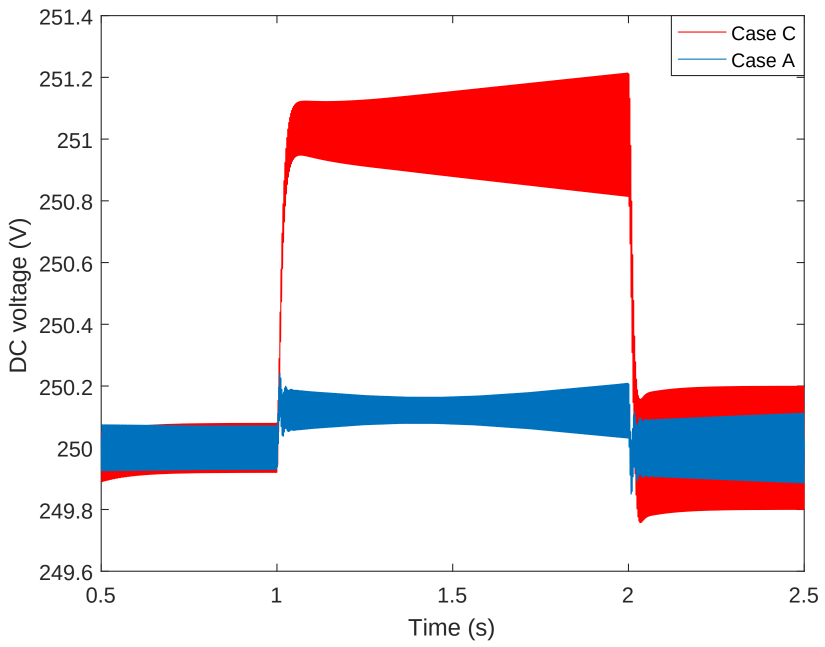

Figure 5.20: Comparison of voltage between case 'A' and case 'C'

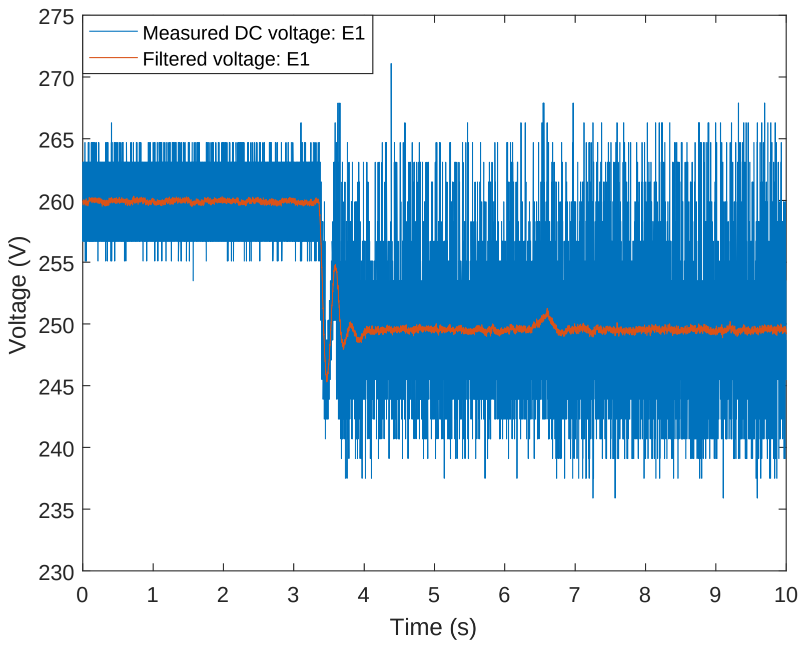

Figure 5.21: Measured DC voltage for 'E1'

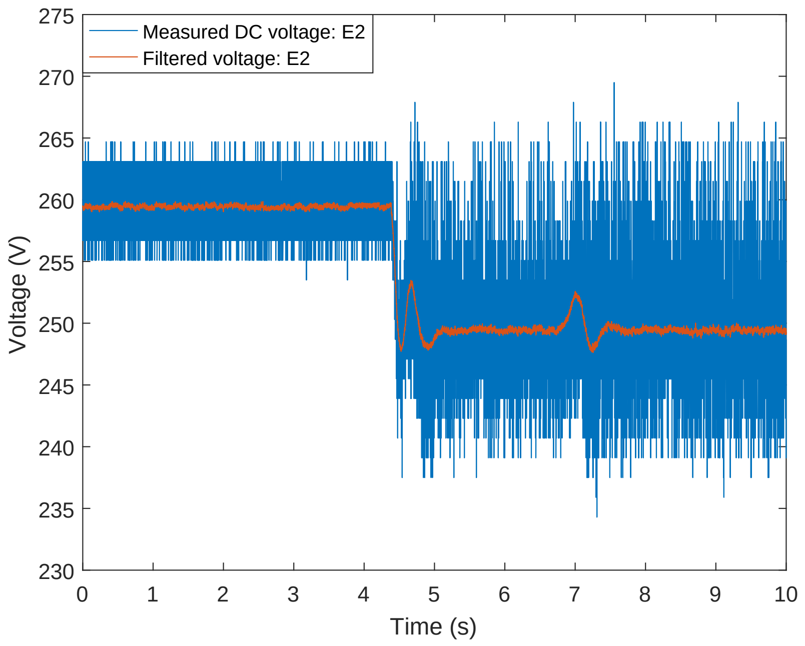

Figure 5.22: Measured DC voltage for 'E2'

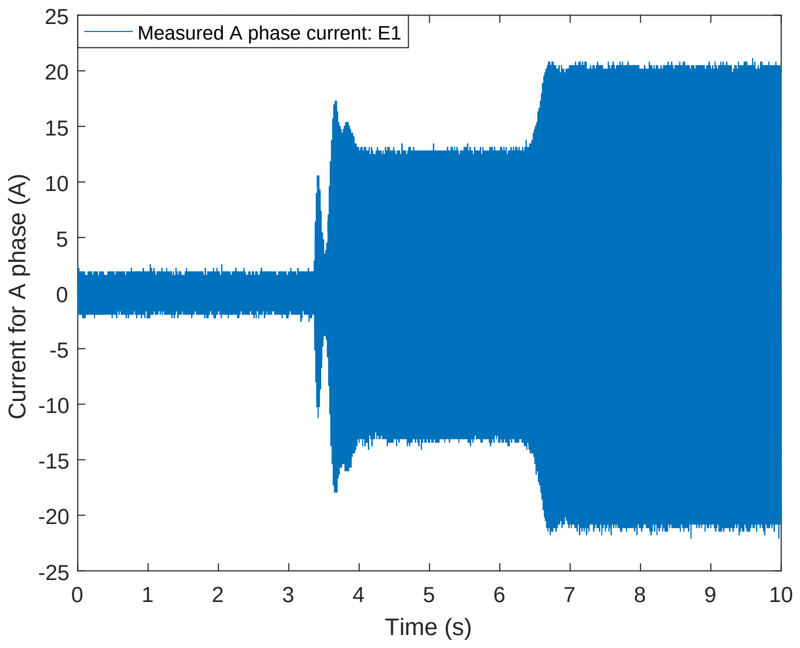

Figure 5.23: Measured A phase current for ‘E1’

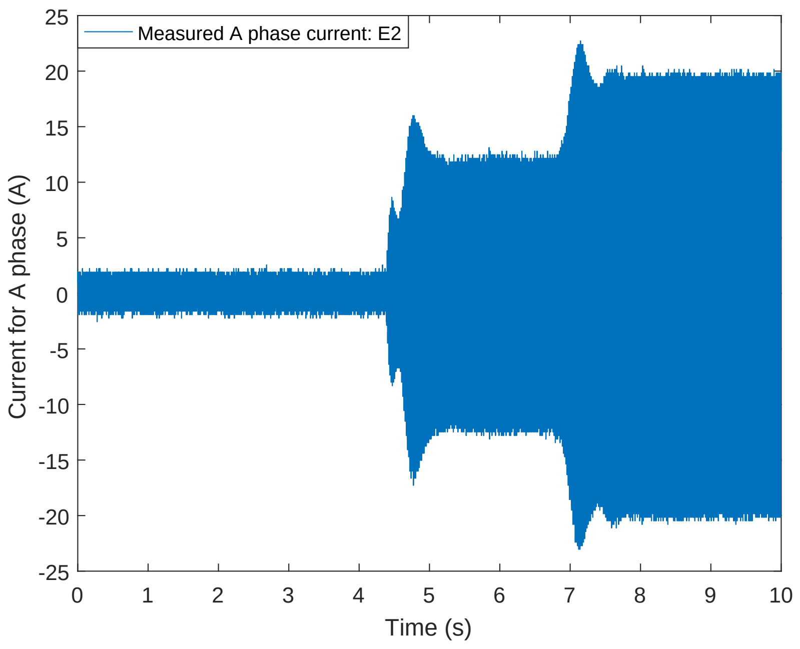

Figure 5.24: Measured A phase current for ‘E2’

CHAPTER 6
CONCLUSIONS AND FUTURE WORK

In this work, the application studies of genetic algorithms with stochastic problems are first studied. Genetic algorithms are widely used to solve practical problems that are difficult or impossible to solve manually. And practical problems are usually stochastic in real life scenarios. Thus there are many researchers seeking ways to improve the efficiency of genetic algorithms.

The first topic of this dissertation is to study the integration of computing budget allocation schemes into the single objective genetic algorithms. The purpose of the study is to find the most efficient method of applying the genetic algorithms in solving single objective stochastic problems. A single objective genetic algorithm often starts with random initialization, followed by loops that contain tournament selection, crossover, mutation, evaluation until termination. The previous studies focused on applying computing budget allocation (CBA) methods, usually the optimal computing budget allocation (OCBA) proposed by Chen et al., into the evaluation and ranking process (EIGA). This work proposed to allocate the CBA techniques directly into the tournament selection operator rather than the fitness evaluation operator (SIGA). This allows the search algorithm to allocate samples as needed in the tournament selection process and improve the accuracy of choosing the best individuals in the tournament selection. The performance comparisons between the EIGA and the SIGA with different CBA methods are performed on benchmark functions under different noise levels. Each experiment is repeated 1600 times to achieve statistical significance tests results in comparisons. It is shown that the SIGA with different CBA methods is consistently achieving more accurate results than the EIGA for the same amount of total computational budget. And the performance of OCBA is not necessarily better than other CBA methods under the same EIGA or SIGA framework.

The second topic of this dissertation is to study the integration of computing budget allocation schemes into the multi-objective genetic algorithms. The purpose of the study is to find the most efficient method of applying the genetic algorithms in solving multi-objective stochastic problems. Unlike the workflow of the single objective genetic algorithm, the multi-objective genetic algorithms often contain recombination of the previous pool and current pool and selecting the next generation pool from the combined pool. This process is called environmental selection, which poses much greater selection pressure than the tournament selection. Methods of integrating CBA methods into the environmental selection or the evaluation procedure have been investigated by previous researchers separately in [75] and [3]. However, there were no comprehensive studies about comparing the difference between these two techniques. In this work, a combination of studies that compares the alternative approaches to integrating CBA methods into the environmental selection (SIGA) and the evaluation procedure (EIGA) of a multi-objective genetic algorithm have been proposed. Repetitive experiments of 1200 times are performed on typical test problems with the different noise level. The tests results, which are interpreted with general distance (GD) and inverse general distance (IGD) metrics, are compared with statistical significance. It is found that the SIGA generally outperforms the EIGA regarding both GD and IGD metrics. And among the CBA methods integrated, the OCBA performs less favorably than the other CBA methods under the same EIGA or SIGA framework.

The last topic of this dissertation aims to apply the proposed SIGE framework to solve a practical stochastic problem in the area of power electronics. A three-phase photovoltaic inverter system with PI control to stabilize the DC side input voltage and PR control to generate the pulse width modulation (PWM) signal is studied.

The inverter system is subject to noise from the randomness of input power and the influence of dynamic grid. The system can produce huge distortions or even instability if the control parameters are not properly tuned. The proposed selection integrated multi-objective genetic algorithm is used to optimize the control parameters with the random cases. The results are validated with hardware-experiment.

Future work may consider adding objectives from frequency domain, such as maximizing the stability phase margin to increase system stability. Other possible future works include applying the proposed algorithms to solve other stochastic optimization problems in power electronics, such as to optimize a DC to DC converter subject to hardware noise by tuning the control parameters, to optimize a three phase rectifier to deliver maximum power while minimizing converter loss, etc. The objectives for those power electronics problems may also include time domain or frequency domain objectives.

Recently, there is a growing interest in the optimization of the smart grid. The concept of smart grid has emerged to solve the modern distribution problems. The smart grid merges the concepts of information technology, customer participation, and other new technologies. It enables the gathering of and communicating of information on both the supplier side and consumer side and improves the efficiency, sustainability, load balancing, reliability, and flexibility of distribution networks [109]. And the genetic algorithms can be applied to improve different aspects of the smart grid. For example, In [110], Ramaswamy et al. propose the reconfiguration of the grid to achieve a few optimizing objectives such as minimal power loss, minimum voltage deviation, etc. A simple GA is integrated to optimize a 16-node test network. Ana Soares proposed scheduling of domestic electric loads to minimize the end user's electricity bill while maintaining the quality of the energy services [111]. Stephan Hutterer et al. proposed evolutionary-algorithm-based control policies for flexible optimal power flow over time [112]. Dominik Egarter uses a genetic algorithm to determine a set of devices for a load curve which enables a regular meter to act as a smart meter [113]. The genetic algorithms with an efficient sampling technique can be applied to solve various smart grid optimization ranging from the cost, efficiency, realizability and availability with the randomness of the distribution network.

Bibliography

[1] F. Busetti, "Genetic algorithms overview," *Retrieved on December*, vol. 1, 2007.

[2] P. Chen, P. Siano, B. Bak-Jensen, and Z. Chen, "Stochastic optimization of wind turbine power factor using stochastic model of wind power," *Sustainable Energy, IEEE Transactions on*, vol. 1, no. 1, pp. 19–29, 2010.

[3] L. H. Lee, E. P. Chew, S. Teng, and Y. Chen, "Multi-objective simulation-based evolutionary algorithm for an aircraft spare parts allocation problem," *European Journal of Operational Research*, vol. 189, no. 2, pp. 476 – 491, 2008.

[4] D. Yan and H. Mukai, "Stochastic discrete optimization," *SIAM Journal on Control and Optimization*, vol. 30, no. 3, pp. 594–612, 1992.

[5] W.-B. Gong, Y.-C. Ho, and W. Zhai, "Stochastic comparison algorithm for discrete optimization with estimation," *SIAM Journal on Optimization*, vol. 10, no. 2, pp. 384–404, 2000.

[6] C. Schmidt, J. Branke, and S. E. Chick, "Integrating techniques from statistical ranking into evolutionary algorithms," in *Applications of Evolutionary Computing.* Springer, 2006, pp. 752–763.

[7] H. Pan, L. Wang, and B. Liu, "Particle swarm optimization for function optimization in noisy environment," *Applied Mathematics and Computation*, vol. 181, no. 2, pp. 908–919, 2006.

[8] T. M. Alkhamis and M. A. Ahmed, "Simulation-based optimization using simulated annealing with confidence interval," in *Proceedings of the 36th conference on Winter simulation.* Winter Simulation Conference, 2004, pp. 514–519.

[9] A. M. Cramer, S. D. Sudhoff, and E. L. Zivi, "Evolutionary algorithms for minimax problems in robust design," *Evolutionary Computation, IEEE Transactions on*, vol. 13, no. 2, pp. 444–453, 2009.

[10] B. Cassimere, R. Chan, J. Cale, A. Cramer, and S. Sudhoff, "Evolutionary design of electromagnetic and electromechanical devices," in *Electric Ship Technologies Symposium, 2007. ESTS '07. IEEE*, May 2007, pp. 150–157.

[11] L. Wang, L. Zhang, and D.-Z. Zheng, "Genetic ordinal optimisation for stochastic flow shop scheduling," *The International Journal of Advanced Manufacturing Technology*, vol. 27, no. 1-2, pp. 166–173, 2005.

[12] C.-H. Chen, D. He, M. Fu, and L. H. Lee, "Efficient simulation budget allocation for selecting an optimal subset," *INFORMS Journal on Computing*, vol. 20, no. 4, pp. 579–595, 2008.

[13] K. Deb, "An introduction to genetic algorithms," *Sadhana*, vol. 24, no. 4-5, pp. 293–315, 1999.

[14] A. Popov, "Genetic algorithms for optimization," *User Manual, Hamburg*, vol. 2013, 2005.

[15] D. E. Goldberg and K. Deb, "A comparative analysis of selection schemes used in genetic algorithms," *Urbana*, vol. 51, pp. 61 801–2996, 1991.

[16] B. L. Miller and D. E. Goldberg, "Genetic algorithms, tournament selection, and the effects of noise," *Complex Systems*, vol. 9, no. 3, pp. 193–212, 1995.

[17] R. B. Agrawal, K. Deb, and R. B. Agrawal, "Simulated binary crossover for continuous search space," 1994.

[18] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: Nsga-ii," *IEEE transactions on evolutionary computation*, vol. 6, no. 2, pp. 182–197, 2002.

[19] M. Abido, "Environmental/economic power dispatch using multiobjective evolutionary algorithms," *IEEE Transactions on Power Systems*, vol. 18, no. 4, pp. 1529–1537, 2003.

[20] J. W. Kolar, J. Biela, and J. Minibock, "Exploring the pareto front of multi-objective single-phase pfc rectifier design optimization-99.2% efficiency vs. 7kw/din 3 power density," in *Power Electronics and Motion Control Conference, 2009. IPEMC'09. IEEE 6th International.* IEEE, 2009, pp. 1–21.

[21] Z. Jiang, L. Gao, and R. A. Dougal, "Flexible multiobjective control of power converter in active hybrid fuel cell/battery power sources," *IEEE Transactions on Power Electronics*, vol. 20, no. 1, pp. 244–253, 2005.

[22] R. T. Marler and J. S. Arora, "Survey of multi-objective optimization methods for engineering," *Structural and multidisciplinary optimization*, vol. 26, no. 6, pp. 369–395, 2004.

[23] E. Zitzler, M. Laumanns, and L. Thiele, "Spea2: Improving the strength pareto evolutionary algorithm for multiobjective optimization," *Evol. Methods for Design, Optimization, and Control*, pp. 95–100, 2002.

[24] C.-H. Chen, J. Lin, E. Yücesan, and S. E. Chick, "Simulation budget allocation for further enhancing the efficiency of ordinal optimization," *Discrete Event Dynamic Systems*, vol. 10, no. 3, pp. 251–270, 2000.

[25] B.-W. Hsieh, C.-H. Chen, and S.-C. Chang, "Efficient simulation-based composition of scheduling policies by integrating ordinal optimization with design of experiment," *Automation Science and Engineering, IEEE Transactions on*, vol. 4, no. 4, pp. 553–568, 2007.

[26] C.-H. Chen, S. D. Wu, and L. Dai, "Ordinal comparison of heuristic algorithms using stochastic optimization," *Robotics and Automation, IEEE Transactions on*, vol. 15, no. 1, pp. 44–56, 1999.

[27] L. Trailovic and L. Y. Pao, "Computing budget allocation for efficient ranking and selection of variances with application to target tracking algorithms," *Automatic Control, IEEE Transactions on*, vol. 49, no. 1, pp. 58–67, Jan 2004.

[28] B.-W. Hsieh, C.-H. Chen, and S.-C. Chang, "Scheduling semiconductor wafer fabrication by using ordinal optimization-based simulation," *Robotics and Automation, IEEE Transactions on*, vol. 17, no. 5, pp. 599–608, 2001.

[29] Y.-C. Ho, R. Sreenivas, and P. Vakili, "Ordinal optimization of deds," *Discrete event dynamic systems*, vol. 2, no. 1, pp. 61–88, 1992.

[30] L. Dai, "Convergence properties of ordinal comparison in the simulation of discrete event dynamic systems," *Journal of Optimization Theory and Applications*, vol. 91, no. 2, pp. 363–388, 1996.

[31] P. Bratley, B. L. Fox, and L. E. Schrage, *A guide to simulation.* Springer, 1983, vol. 2.

[32] S. E. Chick, "Bayesian analysis for simulation input and output," in *Proceedings of the 29th conference on Winter simulation.* IEEE Computer Society, 1997, pp. 253–260.

[33] A. M. Law, W. D. Kelton, and W. D. Kelton, *Simulation modeling and analysis.* McGraw-Hill New York, 1991, vol. 2.

[34] C.-h. Chen, *Stochastic simulation optimization: an optimal computing budget allocation.* World Scientific, 2010, vol. 1.

[35] M. Guidolin and C. Mortarino, "Cross-country diffusion of photovoltaic systems: modelling choices and forecasts for national adoption patterns," *Technological Forecasting and Social Change*, vol. 77, no. 2, pp. 279–296, 2010.

[36] F. Blaabjerg and E. Koutroulis, "Methods for the optimal design of grid-connected pv inverters," *International Journal of Renewable Energy Research (IJRER)*, vol. 1, no. 2, pp. 54–64, 2011.

[37] E. Twining and D. G. Holmes, "Grid current regulation of a three-phase voltage source inverter with an lcl input filter," *IEEE Transactions on Power Electronics*, vol. 18, no. 3, pp. 888–895, 2003.

[38] R. Teodorescu, F. Blaabjerg, M. Liserre, and P. C. Loh, "Proportional-resonant controllers and filters for grid-connected voltage-source converters," *IEE Proceedings-Electric Power Applications*, vol. 153, no. 5, pp. 750–762, 2006.

[39] M. T. Jensen, "Generating robust and flexible job shop schedules using genetic algorithms," *IEEE Trans. Evol. Comput.*, vol. 7, no. 3, pp. 275–288, 2003.

[40] Y. Xu, Z. Y. Dong, K. Meng, J. H. Zhao, and K. P. Wong, "A hybrid method for transient stability-constrained optimal power flow computation," *IEEE Trans. Power App. Syst.*, vol. 27, no. 4, pp. 1769–1777, 2012.

[41] O. Stan, R. Sirdey, J. Carlier, and D. Nace, “The robust binomial approach to chance-constrained optimization problems with application to stochastic partitioning of large process networks,” *J. Heuristics*, vol. 20, no. 3, pp. 261–290, 2014.

[42] S. Buadhachain and G. Provan, “An efficient decentralized clustering algorithm for aggregation of noisy multi-mean data,” *J. Heuristics*, vol. 21, no. 2, pp. 301–328, 2015.

[43] J. Mendoza, L.-M. Rousseau, and J. Villegas, “A hybrid metaheuristic for the vehicle routing problem with stochastic demand and duration constraints,” *J. Heuristics*, pp. 1–28, 2015.

[44] G. K. Dill *et al.*, “Robust design of power system controllers based on optimization of pseudospectral functions,” *IEEE Trans. Power App. Syst.*, vol. 28, no. 2, pp. 1756–1765, 2013.

[45] S. Nesmachnow, “An overview of metaheuristics: accurate and efficient methods for optimisation,” *International Journal of Metaheuristics*, vol. 3, no. 4, pp. 320–347, 2014.

[46] C. Blum and A. Roli, “Metaheuristics in combinatorial optimization: Overview and conceptual comparison,” *ACM Computing Surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

[47] S. Tsutsui and A. Ghosh, “Genetic algorithms with a robust solution searching scheme,” *IEEE Trans. Evol. Comput.*, vol. 1, no. 3, pp. 201–208, 1997.

[48] M. Liu and J. He, “An evolutionary negative-correlation framework for robust analog-circuit design under uncertain faults,” *IEEE Trans. Evol. Comput.*, vol. 17, no. 5, pp. 640–665, 2013.

[49] R. R. Chan and S. D. Sudhoff, “An evolutionary computing approach to robust design in the presence of uncertainties,” *IEEE Trans. Evol. Comput.*, vol. 14, no. 6, pp. 900–912, 2010.

[50] S. L. Ho, S. Yang, Y. Bai, and J. Huang, “A robust metaheuristic combining clonal colony optimization and population-based incremental learning methods,” *IEEE Trans. Magn.*, vol. 50, no. 2, pp. 677–680, 2014.

[51] J. Branke, S. Meisel, and C. Schmidt, “Simulated annealing in the presence of noise,” *J. Heuristics*, vol. 14, no. 6, pp. 627–654, 2008.

[52] J. Rada-Vilela, M. Johnston, and M. Zhang, “Population statistics for particle swarm optimization: Resampling methods in noisy optimization problems,” *Swarm and Evol. Comput.*, vol. 17, pp. 37–59, 2014.

[53] K. Deb, S. Gupta, D. Daum, J. Branke, A. K. Mall, and D. Padmanabhan, "Reliability-based optimization using evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 13, no. 5, pp. 1054–1074, 2009.

[54] Z. Tu and Y. Lu, "A robust stochastic genetic algorithm (stGA) for global numerical optimization," *IEEE Trans. Evol. Comput.*, vol. 8, no. 5, pp. 456–470, 2004.

[55] I. Paenke, J. Branke, and Y. Jin, "Efficient search for robust solutions by means of evolutionary algorithms and fitness approximation," *IEEE Trans. Evol. Comput.*, vol. 10, no. 4, pp. 405–420, 2006.

[56] B. Liu, Q. Zhang, F. V. Fernández, and G. G. Gielen, "An efficient evolutionary algorithm for chance-constrained bi-objective stochastic optimization," *IEEE Trans. Evol. Comput.*, vol. 17, no. 6, pp. 786–796, 2013.

[57] B. Liu, F. V. Fernández, and G. G. Gielen, "Efficient and accurate statistical analog yield optimization and variation-aware circuit sizing based on computational intelligence techniques," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 30, no. 6, pp. 793–805, 2011.

[58] J. Rada-Vilela, M. Zhang, and M. Johnston, "Optimal computing budget allocation in particle swarm optimization," in *Proc. 15th Ann. Conf. Genetic, Evol. Comput.*, 2013, pp. 81–88.

[59] S.-C. Horng and S.-Y. Lin, "Autionary algorithm assisted by surrogate model in the framework of ordinal optimization and optimal computing budget allocation," *Inform. Sci.*, vol. 233, pp. 214–229, 2013.

[60] H. Xiao and L. H. Lee, "Simulation optimization using genetic algorithms with optimal computing budget allocation," *Simulation*, vol. 90, no. 10, pp. 1146–1157, 2014.

[61] Y. Ermoliev, *Numerical techniques for stochastic optimization.* Springer-Verlag New York, Inc., 1988.

[62] M. C. Fu, C.-H. Chen, and L. Shi, "Some topics for simulation optimization," in *Proceedings of the 40th Conference on Winter Simulation.* Winter Simulation Conference, 2008, pp. 27–38.

[63] K. Deb and D. Deb, "Analysing mutation schemes for real-parameter genetic algorithms," *International Journal of Artificial Intelligence and Soft Computing*, vol. 4, no. 1, pp. 1–28, 2014.

[64] J. Liang, B. Qu, and P. Suganthan, "Problem definitions and evaluation criteria for the CEC 2014 special session and competition on single objective real-parameter numerical optimization," *Computational Intelligence Laboratory, Zhengzhou University, Zhengzhou China and Technical Report, Nanyang Technological University, Singapore*, 2013.

[65] M. Siddiqui, "Statistical inference for rayleigh distributions," *J. Res. NBS D Rad. Sci.*, vol. 68, pp. 1005–1010, 1964.

[66] L. H. Lee, C. Chen, E. P. Chew, J. Li, N. A. Pujowidianto, and S. Zhang, "A review of optimal computing budget allocation algorithms for simulation optimization problem," *International J. Operational Res.*, vol. 7, no. 2, pp. 19–31, 2010.

[67] A. M. Cramer, S. D. Sudhoff, and E. L. Zivi, "Metric optimization-based design of systems subject to hostile disruptions," *IEEE Trans. Syst., Man, Cybern. A*, vol. 41, no. 5, pp. 989–1000, 2011.

[68] D. P. Heyman and M. J. Sobel, *Stochastic Models in Operations Research: Stochastic Optimization*. Courier Corporation, 2003.

[69] C. Kahraman, *Fuzzy multi-criteria decision making: theory and applications with recent developments*. Springer Science & Business Media, 2008.

[70] C. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary algorithms for solving multi-objective problems*. Springer Science & Business Media, 2007.

[71] H. Sato, Y. Hasegawa, D. Bollegala, and H. Iba, "Improved sampling using loopy belief propagation for probabilistic model building genetic programming," *Swarm and Evol. Comput.*, vol. 23, pp. 1–10, 2015.

[72] L. H. Lee, E. P. Chew, S. Teng, and D. Goldsman, "Optimal computing budget allocation for multi-objective simulation models," in *Proc. 2004 Winter on Simulation Conf.*, vol. 1, 2004.

[73] L. T. Bui, H. A. Abbass, and D. Essam, "Fitness inheritance for noisy evolutionary multi-objective optimization," in *Proc. 7th Annu. Conf. Genetic and Evol. Comput.*, 2005, pp. 779–785.

[74] H. Eskandari, C. D. Geiger, and R. Bird, "Handling uncertainty in evolutionary multiobjective optimization: Spga," in *IEEE Congr. Evol. Comput.*, 2007, pp. 4130–4137.

[75] A. Syberfeldt, A. Ng, R. I. John, and P. Moore, "Evolutionary optimisation of noisy multi-objective problems using confidence-based dynamic resampling," *Eur. J. Oper. Res.*, vol. 204, no. 3, pp. 533–544, 2010.

[76] P. Boonma and J. Suzuki, "A confidence-based dominance operator in evolutionary algorithms for noisy multiobjective optimization problems," in *21st Int. Conf. Tools with Artificial Intell.*, 2009, pp. 387–394.

[77] E. Zitzler, M. Laumanns, and S. Bleuler, "A tutorial on evolutionary multiobjective optimization," in *Metaheuristics for multiobjective optimisation*, 2004, pp. 3–37.

[78] E. J. Chen and L. H. Lee, "A multi-objective selection procedure of determining a pareto set," *Computers and Operations Research*, vol. 36, no. 6, pp. 1872–1879, 2009.

[79] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, 2000.

[80] K. Deb, L. Thiele, M. Laumanns, and E. Zitzler, *Scalable test problems for evolutionary multiobjective optimization*, 2005.

[81] ——, "Scalable multi-objective optimization test problems," in *Proc. Congr. Evol. Comput. (CEC-2002)*, 2002, pp. 825–830.

[82] T. Goel, R. Vaidyanathan, R. T. Haftka, W. Shyy, N. V. Queipo, and K. Tucker, "Response surface approximation of pareto optimal front in multi-objective optimization," *Comput. Methods Appl. M.*, vol. 196, no. 4, pp. 879–893, 2007.

[83] L. T. Bui, D. Essam, H. A. Abbass, and D. Green, "Performance analysis of evolutionary multi-objective optimization methods in noisy environments," in *Proc. 8th Asia Pacific Symp. Intell. and Evol. Syst.*, 2004, pp. 29–39.

[84] C. K. Goh and K. C. Tan, "An investigation on noisy environments in evolutionary multiobjective optimization," *IEEE Tran. Evol. Comput.*, vol. 11, no. 3, pp. 354–381, 2007.

[85] T. Okabe, Y. Jin, and B. Sendhoff, "A critical survey of performance indices for multi-objective optimisation," in *2003 Congr. Evol. Comput.*, vol. 2, 2003, pp. 878–885.

[86] X. Li, J. Branke, and M. Kirley, "On performance metrics and particle swarm methods for dynamic multiobjective optimization problems," in *2007 Congr. Evol. Comput.*, 2007, pp. 576–583.

[87] G. G. Yen and Z. He, "Performance metric ensemble for multiobjective evolutionary algorithms," *IEEE Tran. Evol. Comput.*, vol. 18, no. 1, pp. 131–144, 2014.

[88] M. Fortunato, A. Giustiniani, G. Petrone, G. Spagnuolo, and M. Vitelli, "Maximum power point tracking in a one-cycle-controlled single-stage photovoltaic inverter," *IEEE Trans. Ind. Electron.*, vol. 55, no. 7, pp. 2684–2693, 2008.

[89] A. M. Cramer, S. D. Sudhoff, and E. L. Zivi, "Performance metrics for electric warship integrated engineering plant battle damage response," *IEEE Trans. Aero. Elec. Sys.*, vol. 47, no. 1, pp. 634–646, 2011.

[90] H.-C. Chen, L. Dai, C.-H. Chen, and E. Yücesan, "New development of optimal computing budget allocation for discrete event simulation," in *Proceedings of the 29th conference on Winter simulation.* IEEE Computer Society, 1997, pp. 334–341.

[91] C.-H. Chen, "An effective approach to smartly allocate computing budget for discrete event simulation," in *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, vol. 3. IEEE, 1995, pp. 2598–2603.

[92] C.-H. Chen, Y. Yuan, H.-C. Chen, E. Yucesan, and L. Dai, "Computing budget allocation for simulation experiments with different system structures," in *Simulation Conference Proceedings, 1998. Winter*, vol. 1. IEEE, 1998, pp. 735–741.

[93] T. Shimizu, O. Hashimoto, and G. Kimura, "A novel high-performance utility-interactive photovoltaic inverter system," *Power Electronics, IEEE Transactions on*, vol. 18, no. 2, pp. 704–711, 2003.

[94] F. Schimpf, L. E. Norum *et al.*, "Grid connected converters for photovoltaic, state of the art, ideas for improvement of transformerless inverters," in *Nordic Workshop on Power and Industrial Electronics (NORPIE/2008), June 9-11, 2008, Espoo, Finland.* Helsinki University of Technology, 2008.

[95] M. Calais, V. G. Agelidis, and M. Meinhardt, "Multilevel converters for single-phase grid connected photovoltaic systems: an overview," *Solar Energy*, vol. 66, no. 5, pp. 325–335, 1999.

[96] W. D. Pesnell, *Solar Dynamics Observatory (SDO).* Cham: Springer International Publishing, 2015, pp. 179–196.

[97] Y. Zhang and Y. W. Li, "Grid harmonics compensation using high-power pwm converters based on combination approach," *IEEE Journal of Emerging and Selected Topics in Power Electronics*, vol. 4, no. 1, pp. 186–197, 2016.

[98] J.-W. Jung and M. J. Hawksford, "An oversampled digital pwm linearization technique for digital-to-analog conversion," *IEEE Transactions on Circuits and Systems Part 1: Regular Papers*, vol. 51, no. 9, pp. 1781–1789, 2004.

[99] R. Boudreaux, R. Nelms, and J. Y. Hung, "Simulation and modeling of a dc-dc converter controlled by an 8-bit microcontroller," in *Applied Power Electronics Conference and Exposition, 1997. APEC'97 Conference Proceedings 1997., Twelfth Annual*, vol. 2. IEEE, 1997, pp. 963–969.

[100] J. Lettl, J. Bauer, and L. Linhart, "Comparison of different filter types for grid connected inverter," in *PIERS Proceedings*, 2011, pp. 1426–1429.

[101] N. Zhang, H. Tang, and C. Yao, "A systematic method for designing a pr controller and active damping of the lcl filter for single-phase grid-connected pv inverters," *Energies*, vol. 7, no. 6, pp. 3934–3954, 2014.

[102] A. Mellit and A. M. Pavan, "A 24-h forecast of solar irradiance using artificial neural network: Application for performance prediction of a grid-connected pv plant at trieste, italy," *Solar Energy*, vol. 84, no. 5, pp. 807–821, 2010.

[103] G. Kopp and J. L. Lean, "A new, lower value of total solar irradiance: Evidence and climate significance," *Geophysical Research Letters*, vol. 38, no. 1, 2011.

[104] C. W. Richardson, "Stochastic simulation of daily precipitation, temperature, and solar radiation," *Water Resources Research*, vol. 17, no. 1, pp. 182–190, 1981.

[105] T. Soubdhan, R. Emilion, and R. Calif, "Classification of daily solar radiation distributions using a mixture of dirichlet distributions," *Solar energy*, vol. 83, no. 7, pp. 1056–1063, 2009.

[106] F. De la Rosa, *Harmonics and power systems.* CRC press, 2006.

[107] L. G. Franquelo, J. Napoles, R. C. P. Guisado, J. I. León, and M. A. Aguirre, "A flexible selective harmonic mitigation technique to meet grid codes in three-level pwm converters," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 6, pp. 3022–3029, 2007.

[108] M. Castilla, J. Miret, J. Matas, L. G. de Vicuña, and J. M. Guerrero, "Control design guidelines for single-phase grid-connected photovoltaic inverters with damped resonant harmonic compensators," *IEEE Transactions on industrial electronics*, vol. 56, no. 11, pp. 4492–4501, 2009.

[109] R. Hassan and G. Radman, "Survey on smart grid," in *IEEE SoutheastCon 2010 (SoutheastCon), Proceedings of the*, March 2010, pp. 210–213.

[110] P. Ramaswamy and G. Deconinck, "Smart grid reconfiguration using simple genetic algorithm and nsga-ii," in *Innovative Smart Grid Technologies (ISGT Europe), 2012 3rd IEEE PES International Conference and Exhibition on*, Oct 2012, pp. 1–8.

[111] A. Soares, A. Gomes, C. Henggeler Antunes, and H. Cardoso, "Domestic load scheduling using genetic algorithms," in *Proceedings of the 16th European Conference on Applications of Evolutionary Computation*, ser. EvoApplications'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 142–151.

[112] S. Hutterer, M. Affenzeller, and F. Auinger, "Evolutionary algorithm based control policies for flexible optimal power flow over time," in *Proceedings of the 16th European Conference on Applications of Evolutionary Computation*, ser. EvoApplications'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 152–161.

[113] D. Egarter, A. Sobe, and W. Elmenreich, "Evolving non-intrusive load monitoring," in *Proceedings of the 16th European Conference on Applications of Evolutionary Computation*, ser. EvoApplications'13. Berlin, Heidelberg: Springer-Verlag, 2013, pp. 182–191.

VITA

Mengmei Liu was born in Jingzhou, Hubei Province, P.R. China.

Education

Wuhan University of Technology, B.S. awarded June 2010

University of Kentucky, M.S. awarded March 2013

Research Interest

Stochastic optimization, heuristic algorithms, power electronics