6

Robust Multi-Objective Genetic Algorithm (RMOGA) with Online Approximation under Interval Uncertainty

Weiwei Hu[1], *Adeel Butt*[2], *Ali Almansoori*[2], *Shapour Azarm*[1] *and Ali Elkamel*[2,3]
[1]*University of Maryland, College Park, USA*
[2]*Department of Chemical Engineering, The Petroleum Institute, Abu Dhabi, UAE*
[3]*Department of Chemical Engineering, University of Waterloo, Canada*

6.1 Introduction

Chemical process design and operations often require optimization of conflicting objectives subject to multiple constraints (e.g., Al-Sharrah *et al.*, 2001; Halsallwhitney and Thibaut, 2006; Rangaiah, 2009). For example, to minimize the utility cost and maximize the product output while ensuring the quality of the end products. Such problems are identified as multi-objective optimization (MOO) (Cohon, 1978; Ehrgott, 2005). Recently the use of nature-inspired approaches based on genetic algorithms (Holland, 1975), simulated annealing (Kirkpatrick *et al.*, 1983), and particle swarm algorithm (Kennedy and Eberhart, 1995) have drawn increasing interest in the area of MOO. A common characteristic in these approaches is that instead of searching for one optimum solution, a so-called Pareto-optimum solution set can be obtained in ‘one go’ of the optimization algorithm (Deb, 2001). Many population-based multi-objective optimization approaches (e.g., Goldberg, 1989; Serafini, 1992; Coello Coello, 2002) use meta-heuristics to obtain an estimate of the Pareto set (Evans *et al.*, 1991;

Multi-Objective Optimization in Chemical Engineering: Developments and Applications, First Edition.
Edited by Gade Pandu Rangaiah and Adrián Bonilla-Petriciolet.

Deb, 2007; Ferreira *et al.*, 2008), among which the Multi-objective Genetic Algorithm (MOGA) has become quite popular (Fonseca and Fleming, 1993).

One concern in using MOGA for chemical process optimization is that, often, system inputs (variables and/or parameters) may be uncertain (Limbourg, 2005; Basseur and Zitzler, 2006). For example, the temperature in a distillation column may be fluctuating in an uncertain way around a nominal value. Consequently, such uncertainty can create uncertain variations in the system outputs or objective and/or constraint functions of the model. Since deterministic optimum solutions are typically located on or close to the boundary of one or more constraint functions, a small variation in the inputs can lead to a constraint violation and potentially results in a failure of the system (Nikolaidis *et al.*, 2004). Such a problem can be addressed by using the concept of a robust design. Established by Taguchi, for a robust (or insensitive) design, variations in its response (objective and/or constraint function) remain acceptable (Taguchi, 1987) for all realizations of input uncertainty. Applying the robust design concept during the course of optimization, also called robust optimization, one is able to obtain solutions that are not only optimum but also relatively insensitive (in terms of objective and constraint functions) to uncertainties (e.g., Deb and Gupta, 2005; Gunnawan and Azarm, 2005; Li *et al.*, 2006; Ferreira *et al.*, 2008).

Different approximation (metamodeling) techniques are used in the literature, for example artificial neural network (Mitra and Majumder, 2011), radial basis function (Ray *et al.*, 2009), Kriging (Voutchkov and Keane, 2010), among others, for solving multi-objective optimization problems. The purpose of the approximation is to replace a potentially computationally expensive objective/constraint function with an inexpensive metamodel or surrogate. To construct a metamodel, a set of sample points needs to be determined, which is called design of experiment (Sacks *et al.*, 1989). Approximation techniques for multi-objective optimization can be either offline (Ray *et al.*, 2009) or online (Voutchkov and Keane, 2010). The main difference between these two is that sample points in an offline technique are not updated during optimization while the sample points in an online technique are updated.

In this chapter, two approaches in RMOGA—nested RMOGA and sequential RMOGA—are presented. In RMOGA approaches, objective and feasibility robustness measures are evaluated based on a worst-case analysis. The nested RMOGA was originally developed by Li *et al.* (2006), and included a computationally intensive two-level (nested) solution approach. To address its computational difficulty, the nested RMOGA is improved with a sequential approach (Hu *et al.*, 2011). It is shown that the sequential RMOGA requires a significantly less number of function calls than the nested RMOGA. However, both nested and sequential RMOGA have some limitations, which can be addressed, including: (i) In both RMOGAs (Li *et al.* 2006; Hu *et al.*, 2011), objective robustness requires that the variations (both increasing and decreasing values) in objective functions remain within an acceptable range. However, in this chapter, the objective robustness is considered with a one-sided variation because in chemical application, the decision maker is typically indifferent with a variation in the objective function as long as such variation does not degrade the expected performance. For example, a downside variation in a utility cost implies a cost reduction, which is desirable. (ii) The nested RMOGA is computationally expensive and cannot be applied efficiently for chemical applications. An online approximation technique is therefore integrated with the nested RMOGA in this chapter. In online approximation, sample points are selected online based on intermediate solutions and used to improve the accuracy of the metamodel. The optimum solutions are verified based on the mean squared error in the true and estimated objective and constraint function values to ensure they are acceptable. The procedures for the nested and sequential RMOGA with online approximation are presented and compared. Two case studies (one numerical and the other based on a petroleum refinery) are used to show the applicability of the approaches in RMOGA. It is shown that, without using an approximation technique, the optimum solutions from both RMOGAs are consistent but require a large number of function calls. However, by using an approximation technique, both RMOGA techniques are able to arrive at the optimum solutions using a reasonably small number of function calls.

In section 6.2, an overview of related background and definitions is provided. In Section 6.3, two RMOGAs are presented and compared with a corresponding online approximation method that is combined with these two techniques. Section 6.4 demonstrates RMOGAs with two examples. Finally, a conclusion is provided in section 6.5.

6.2 Background and Definition

A general formulation for a multi-objective optimization problem is defined in Equation 6.1:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f}(\mathbf{x}, \mathbf{p}) \\ \text{s.t} \quad & \mathbf{g}(\mathbf{x}, \mathbf{p}) \le 0 \\ & \mathbf{x} \in [\mathbf{x}^l, \mathbf{x}^u] \end{aligned} \tag{6.1}$$

In this formulation, $\mathbf{x} = (x_1, x_2, \ldots, x_i)$ is a vector of i variables and $\mathbf{p} = (p_1, p_2, \ldots, p_j)$ is a vector of j parameters. $\mathbf{f} = (f_1, f_2, \ldots, f_m)$ represents an objective function vector and $\mathbf{g} = (g_1, g_2, \ldots, g_n)$ represents an inequality constraint vector. In Equation 6.1, the superscripts l and u in variable $\mathbf{x}$ represent the lower and upper bounds, respectively and, while the variables can be changed by an optimizer, parameters are fixed during an optimization run. However, $\mathbf{p}$ (and even $\mathbf{x}$) can have uncertainty (see section 2.2).

The feasible domain in Equation 6.1, denoted by Ω, consists of the set of points that satisfy all constraints. For a minimization problem, as in Equation 6.1, a point $\mathbf{x}_1$ is said to multi-objectively dominate $\mathbf{x}_2$, if $\mathbf{f}(\mathbf{x}_1) \le \mathbf{f}(\mathbf{x}_2)$ for all objective functions with strict inequality holding for at least one objective function (Miettinen, 1999). A solution point $\mathbf{x} \in \Omega$ is non-dominated if there does not exist another solution point $\mathbf{y} \in \Omega$ that dominates it. A non-dominated set Ψ, or Pareto solution set (Pareto frontier), is defined as: $\{\mathbf{x} \in \Psi |$ there does not exist $\mathbf{y} \in \Omega$ such that $\mathbf{y}$ dominates $\mathbf{x}\}$.

One way to measure the relative goodness of Pareto frontier is by using the quality metrics (Wu and Azarm, 2001). The Hyperarea Difference (HD) and Overall Spread (OS) are the two quality metrics used in this study based on a set of non-dominated points. As shown in Figure 6.1, HD is represented by the shaded area based on a definition of a good point (P_{good}) and a bad point (P_{bad}) in the objective space. As HD measures the closeness of the non-dominated points to a good point, the smaller the HD value the better. On the other hand, OS is defined as the ratio between the rectangle area bounded by the two extreme points of the non-dominated points {a-f-e-m} to the rectangle area bounded by the good and bad points. Since OS measures the spread of the set of non-dominated points, the larger the value of OS is the better the spread of the non-dominated points.

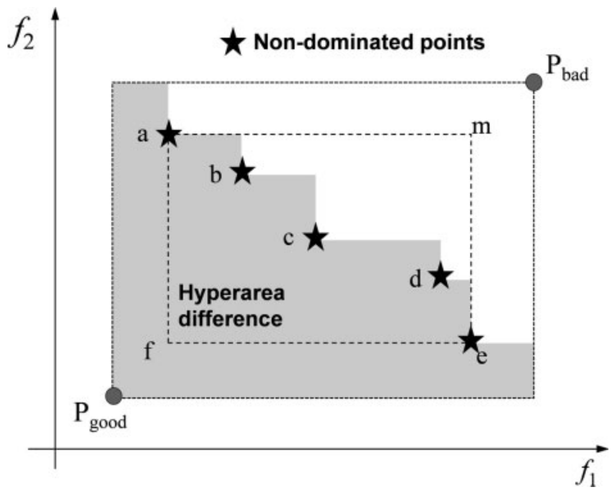

Figure 6.1 *Quality metrics for a set of non-dominated points.*

The classical approach for obtaining a non-dominated set for Equation 6.1 is to select one design objective that is most important as the main objective subject to the other objectives as constraints (e.g., Ignizio, 1978). An alternative approach is to combine all design objectives into a single function with a weighted summation. This approach uses various weights to generate different non-dominated points but suffers from the difficulty in identifying the entire Pareto frontier (Miettinen, 1999). These methods obtain one Pareto solution with each optimization run. Instead of obtaining one solution point at a time, a population-based approach, such as a genetic algorithm, can be combined with a non-dominated sorting method in order to obtain all the solution points, as detailed below. The other advantage of GA is that it can solve highly nonlinear optimization problems with continuous and/or discrete variables and also can obtain a global optimum solution.

6.2.1 Multi-Objective Genetic Algorithm (MOGA)

The MOGA is a population based multi-objective optimization approach first introduced by Fonseca and Fleming (1993). Due to its meta-heuristic nature, MOGA is capable of obtaining a good estimate of the Pareto frontier. A good estimate of the Pareto frontier is one that covers a large portion of the true Pareto front. MOGA operates on a population of points. Each point defines a chromosome which can be in the form of either a combination of real-valued variables (real-coded GA) or a binary string (binary-coded GA). The fitness of each point is a measure of performance of the point as defined by the objective and constraint functions. If a point violates constraints, the value of the objective function is penalized. MOGA basically consists of three parts: (i) coding and decoding the points into chromosome; (ii) evaluating the fitness of each point; and (iii) applying genetic operators to generate the next generation of points. Among these three parts, the first and third parts in MOGA are essentially the same as those in a GA (Goldberg, 1989).

The fitness of each point in MOGA is evaluated by performing a sorting algorithm based on the value of the objective functions. A commonly used sorting algorithm is Non-Dominated Sorting (NDS), see, for example, Deb (2001). The NDS procedure, based on a population of strings (individuals or points), works as follows. First, a non-dominated set Φ_1 among a population of strings is determined based on their objective function values. All members in Φ_1 are assigned to the first-rank points. From the remaining points, the set of non-dominated points Φ_2 is detected and its members are assigned to the second rank. This procedure is repeated, until the whole population is divided into partitions $\Phi_1, \Phi_2, \ldots, \Phi_s$. Members in each of these partitions are assigned to ranks $1, 2, \ldots, s$. Obviously, there can be more than one element in each of the partitions. In order to establish a distinctive ranking among the elements of a particular partition, crowding distance sorting can be used (Coello Coello *et al.*, 2002). Using a crowding distance sorting, individuals that contribute more to the diversity are assigned higher ranks. In addition to NDS, other sorting schemes (Knowles and Corne, 1999) can also be used for fitness assignment in MOGA.

A simple penalty method can be used in MOGA to handle constrained multi-objective optimization problems. Using a penalty method, in minimization problems, the infeasible points are penalized by adding a large positive value to their original fitness values. Typically, the penalty value is proportional to their constraint function values. Since the constraint values for infeasible points are always positive, highly infeasible points with higher constraint values are penalized more than the less infeasible points with smaller constraint values. Other constraint handling techniques (e.g., Kurpati *et al.*, 2002; Qu and Suganthan, 2011) can also be used in MOGA. In general, these constraint handling approaches tend to consider constraint function values during the fitness assignment stage.

6.2.2 Multi-Objective Robustness with Interval Uncertainty: Basic Idea

It is assumed that, at a candidate point, variables **x** and parameters **p** have nominal values. Let $\Delta\mathbf{x}$ and $\Delta\mathbf{p}$ represent the interval uncertainty around the nominal **x** and **p**, respectively. This uncertainty range is prespecified, such as $\Delta\mathbf{x} \in [-\Delta\mathbf{x}^l, \Delta\mathbf{x}^u]$ and $\Delta\mathbf{p} \in [-\Delta\mathbf{p}^l, \Delta\mathbf{p}^u]$. Due to this uncertainty, the value for the objective and constraint functions is changed from their nominal. Typically, for minimization problems, it is undesirable for an objective function value to increase. Let $\Delta\mathbf{f}^+$ denotes an increase in the value of objective functions:

$$\begin{aligned}\Delta\mathbf{f}^+ &= [\mathbf{f}(\mathbf{x}+\Delta\mathbf{x}, \mathbf{p}+\Delta\mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p})]^+ \\ &= \begin{cases} \mathbf{0}, & \text{if } \mathbf{f}(\mathbf{x}+\Delta\mathbf{x}, \mathbf{p}+\Delta\mathbf{p}) \le \mathbf{f}(\mathbf{x}, \mathbf{p}) \\ \mathbf{f}(\mathbf{x}+\Delta\mathbf{x}, \mathbf{p}+\Delta\mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p}), & \text{otherwise} \end{cases}\end{aligned} \tag{6.2}$$

where $\mathbf{f}(\mathbf{x}+\Delta\mathbf{x}, \mathbf{p}^+\Delta\mathbf{p})$ and $\mathbf{f}(\mathbf{x}, \mathbf{p})$ represent the actual and nominal values for the objective functions. $\Delta\mathbf{f}^+$ is called the objective variation vector and its elements represent the increase (if any) in each objective function as a result of input uncertainty. To measure the variation in all objectives with a scalar (single value), the Euclidean norm $\|\Delta\mathbf{f}^+\|$ is used, i.e., $\|\Delta\mathbf{f}^+\| = (\sum_{mi=1}^{m} \Delta f_{mi}^2)^{1/2}$, where *mi* is the index for the objective functions. In solving a multi-objective optimization problem under interval uncertainty, it is important to obtain solutions which are not only optimal but also have acceptable increases in all of their objectives. To achieve this goal, the decision maker can prespecify a positive scalar value

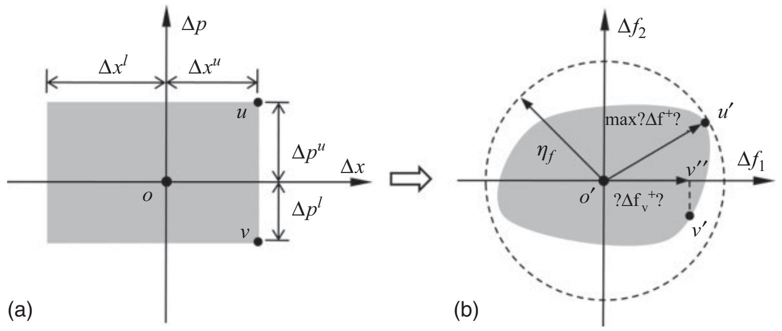

Figure 6.2 *Multi-objective robustness hypothetical case: (a) uncertainty space (b) objective variation space.*

of η_f as the acceptable limit for the objective variations, such that the maximum Euclidean norm as defined above is smaller than or equal to η_f:

$$\max_{\Delta \mathbf{x}, \Delta \mathbf{p}} \left\| \Delta \mathbf{f}^{+} \right\| \leq n_f \tag{6.3}$$

The inequality constraint in Equation 6.3 is referred to as multi-objective robustness constraint. Any candidate point that satisfies Equation 6.3 is considered to be a multi-objective robust point. A two-diemnsional conceptual representation of multi-objective robustness with interval uncertainty is shown in Figure 6.2. The uncertainty space in Figure 6.2(a) has two axes defined by Δx and Δp, where the grey area represent the known uncertain interval. Any point inside the grey area in the uncertainty space corresponds to a realization of uncertainty. For example, as shown in Figure 6.2(a), the point o represents the nominal point ($\mathbf{x}$, $\mathbf{p}$), and point u, v each represents a realized uncertain value. The entire gray area in Figure 6.2(a) can be mapped to the objective variation space in Figure 6.2(b) where the nominal point o' represents the nominal value for objectives, i.e., $\mathbf{f}(\mathbf{x}, \mathbf{p})$. It can be seen that the maximum Euclidean norm of the objective variation vector $\Delta \mathbf{f}^{+}$ is the distance from the nominal point to the furthest point u' on the mapped objective variation range in Figure 6.2(b). Notice that the point u in Figure 6.2(a) is the corresponding point to point u' in Figure 6.2(b). The dash-lined circle in Figure 6.2(b) denotes the acceptable variation range whose radius is equal to η_f.

It should be mentioned that since we are only concerned with the increase in the objectives, any point in the third quadrant in the mapped objective variation range in Figure 6.2(b) (where all objective values are decreased) is unimportant and ignored. However, to calculate the Euclidean norm of the objective variation for points in the second and forth quadrant, those points should be projected first to the positive axis as in Figure 6.2(b). For example, point v'' is the projection of point v' in Figure 6.2(b), while the corresponding point for point v' is point v in Figure 6.2(a). The Euclidean norm of the objective variation for point v, i.e., $\|\Delta \mathbf{f}_v^{+}\|$, is represented by the horizontal line segment along the Δf_1 axis. Likewise, all other points in the second and forth quadrant of the mapped objective variation range (gray area) in Figure 6.2(b) can be transformed. In this way, in searching for the furthest point or the maximum Euclidean norm of the objective variation vector, only the first quadrant (including the transformed points) needs to be considered. Figure 6.2 shows the case in which the furthest point on the mapped objective variation range is located in the first quadrant in the objective variation space. Since the point is within the dashed circle and $\max \|\Delta \mathbf{f}^{+}\| \leq \eta_f$ is satisfied, the candidate point is said to be multi-objectively robust. It should be noted that the Euclidean norm is maximized with respect to $\Delta\mathbf{x}$ and $\Delta\mathbf{p}$. Graphically this means searching for a point in the uncertain interval (gray area) in Figure 6.2(a), which corresponds to the furthest points (point u) mapped into the first quadrant in the objective variation space in Figure 6.2(b). This is essentially a worst case approach to ensure the robustness of a solution point.

Feasibility robustness is defined in a similar way to the definition of multi-objective robustness, with the following inequality constraint:

$$\max_{\Delta\mathbf{x},\Delta\mathbf{p}} \left[\max_{n} \mathbf{g}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) \right] \leq 0 \tag{6.4}$$

In this formulation, the outer "max" is for different realizations of uncertainty $\Delta\mathbf{x}$ and $\Delta\mathbf{p}$, while the inner "max" is with respect to different elements of the constraint functions, $\mathbf{g} = (g_1, g_2, \ldots, g_n)$. Note that for feasibility robustness, the decision maker is only concerned with the feasibility of a candidate point (i.e., $\mathbf{g} \leq \mathbf{0}$), under all realizations of uncertainty. As shown by the feasibility robustness constraint, Equation 6.4, the left-hand side represents the worst case constraint value, which should be less than or equal to zero in order to ensure feasibility. Consequently, any point that satisfies Equation 6.4 is called a feasibly robust point.

As a final note, both objective and feasibility robustness have been defined earlier (Gunawan and Azarm, 2005; Li *et al.*, 2006; Hu *et al.*, 2011) for the case when both positive and negative variations in the objective functions and constraints are considered. In this way, the variation range must be symmetric for both objective and constraint functions. However, this symmetric assumption for the variation range can be relaxed to handle asymmetric variations by specifying two different values of η_f in Equation 6.3. In the next section, the objective and feasibility robustness are integrated within two MOGAs.

6.3 Robust Multi-Objective Genetic Algorithm (RMOGA)

This section presents two approaches in RMOGA for solving a multi-objective optimization problem with interval uncertainty. In both RMOGAs, robustness of a solution point is evaluated based on a worst case analysis. Briefly, the difference between the two approaches is based on when robustness evaluation is carried out. In the nested RMOGA, the lower level subproblems evaluate robustness of all candidate solution points; while in the sequential RMOGA, the robustness of each optimal solution point obtained in the upper level problem. The details of the nested RMOGA (section 6.3.1) and sequential RMOGA (section 6.3.2) are presented next.

6.3.1 Nested RMOGA

The nested RMORO is formulated as a bi-level optimization framework with an upper level problem and two lower level subproblems (Li *et al.*, 2006). In the upper level, MOGA searches the variable space to identify point $\mathbf{x}$ which optimizes the objectives; while in the lower level, a single-objective GA evaluates multi-objective and feasibility robustness of each intermediate point considered in the upper-level. The formulations for the upper- and lower-level problems are given as follows:

Upper level problem:

$$\begin{aligned} \max_{\mathbf{x}} \quad & \mathbf{f}(\mathbf{x}, \mathbf{p}) \\ \text{s.t.} \quad & \mathbf{g}(\mathbf{x}, \mathbf{p}) \le 0 \\ & \max \left\| \Delta \mathbf{f}^{+} \right\| \le \eta_f \\ & \max[\max \mathbf{g}] \le 0 \\ & \mathbf{x} \in [\mathbf{x}^l, \mathbf{x}^u] \end{aligned} \tag{6.5}$$

Lower level subproblems:

$$\begin{aligned} & \max \left\| \Delta \mathbf{f}^{+} \right\| = \max_{\Delta \mathbf{x}, \Delta \mathbf{p}} \left\| [\mathbf{f}(\mathbf{x} + \Delta \mathbf{x}, \mathbf{p} + \Delta \mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p})]^{+} \right\| \\ & \text{s.t. } \Delta \mathbf{x} \in [-\Delta \mathbf{x}^l, \Delta \mathbf{x}^u], \ \Delta \mathbf{p} \in [-\Delta \mathbf{p}^l, \Delta \mathbf{p}^u] \end{aligned} \tag{6.6}$$

$$\begin{aligned} & \max[\max \mathbf{g}] = \max_{\Delta \mathbf{x}, \Delta \mathbf{p}} [\max_{n} \mathbf{g}(\mathbf{x} + \Delta \mathbf{x}, \mathbf{p} + \Delta \mathbf{p})] \\ & \text{s.t. } \Delta \mathbf{x} \in [-\Delta \mathbf{x}^l, \Delta \mathbf{x}^u], \ \Delta \mathbf{p} \in [-\Delta \mathbf{p}^l, \Delta \mathbf{p}^u] \end{aligned} \tag{6.7}$$

The upper level is formulated as a multi-objective optimization problem as in Equation 6.1, except that the multi-objective and feasibility robustness constraints, as defined in Section 6.2.2 are added. Notice that in Equation 6.5 the left-hand side of the inequalities of multi-objective and feasibility robustness constraints, i.e., $\max \|\Delta \mathbf{f}^{+}\|$ and $\max[\max \mathbf{g}]$ must be evaluated in the lower-level sub-problems. As shown in Equation 6.6 and Equation 6.7, the lower level includes two single-objective optimization subproblems where the value for the nominal design denoted by $\mathbf{x}$ is fixed. Essentially, the first optimization subproblem obtains the maximum Euclidean norms of increase in the objective vector, i.e., $\max \|\Delta \mathbf{f}^{+}\|$ and the second optimization subproblems obtains the worst case constraint value, i.e., $\max[\max \mathbf{g}]$.

The nested RMOGA works as follows: MOGA first generates a population of points in the upper level problem. In order to assign fitness to each point, MOGA needs to calculate the value of objective and constraint functions. Since the multi-objective and feasibility robustness constraint in Equation 6.5 needs to be evaluated in the lower level subproblems, MOGA must forward the nominal value of a current point (from the population), as denoted by a vector $\mathbf{x}$, to the lower-level. Once the lower-level sub-problems receive $\mathbf{x}$, the optimization problems in Equation 6.6 and Equation 6.7 are solved with respect to $\Delta \mathbf{x}$ and $\Delta \mathbf{p}$ using a single-objective GA. The optimal value from the two subproblems, i.e., $\max \|\Delta \mathbf{f}^{+}\|$ and $\max[\max \mathbf{g}]$ are obtained and returned to the upper level problem to complete constraint evaluation for the current point $\mathbf{x}$. The same procedure (solving the two lower-level subproblems) is repeated for all other points in the MOGA population. Next, MOGA ranks each point in the current population based on the values of objective and constraint functions (including multi-objective and feasibility robustness constraints). Finally, the fitness of each point is determined, and this completes one MOGA generation. Subsequently, MOGA continues with many generations until some stopping criteria have been satisfied. Typical stopping criteria in the nested RMOGA are: (i) maximum number of generations or function calls is completed; (ii) no improvement in the Pareto solutions from one generation to the next is obtained. Note that in MOGA as well as in RMOGA, one function call refers to a single instance of evaluating the optimization model—that is, a single instance of evaluating together all objective and constraint functions.

The RMOGA presented in this section is called 'nested' because the two lower level subproblems are nested within an upper level problem. With a population-based approach used in the nested RMOGA, the computational effort required by the nested approach grows exponentially as the number of points in the population increases, which is the case when the number of variables increases. In the next section, a sequential RMOGA is presented, which is more efficient than the nested RMOGA approach.

6.3.2 Sequential RMOGA

The sequential RMOGA (Hu *et al.*, 2011) is developed to improve the computational efficiency of the nested RMOGA. This approach is iterative and each iteration involves two steps. In the first step, a deterministic multi-objective optimization problem is solved to obtain a set of optimal solutions; while in the second step, the robustness is evaluated for each optimal solution obtained from the first step. These two steps are alternated iteratively to obtain the robust optimum solutions. The formulation for the optimization problems in the two steps is given in the equations below:

First-step problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & \mathbf{f}(\mathbf{x}, \mathbf{p}) \\ \text{s.t.} \quad & \left\| [\mathbf{f}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p})]^{+} \right\| \le \eta_f, \forall \Delta\mathbf{x}, \Delta\mathbf{p} \in S_f \\ & \mathbf{g}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) \le 0, \forall \Delta\mathbf{x}, \Delta\mathbf{p} \in S_g \\ & \mathbf{x} \in [\mathbf{x}^l, \mathbf{x}^u] \end{aligned} \tag{6.8}$$

Second-step problems:

$$\begin{aligned} & \max_{\Delta\mathbf{x}, \Delta\mathbf{p}} \left\| [\mathbf{f}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p})^{+}] \right\| \le \eta_f \\ & \text{s.t } \Delta\mathbf{x} \in [-\Delta\mathbf{x}^l, \Delta\mathbf{x}^u], \Delta\mathbf{p} \in [-\Delta\mathbf{p}^l, \Delta\mathbf{p}^u] \end{aligned} \tag{6.9}$$

$$\begin{aligned} & \max_{\Delta\mathbf{x}, \Delta\mathbf{p}} [\max_{n} \mathbf{g}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}] \le 0 \\ & \text{s.t. } \Delta\mathbf{x} \in [-\Delta\mathbf{x}^l, \Delta\mathbf{x}^u], \Delta\mathbf{p} \in [-\Delta\mathbf{p}^l, \Delta\mathbf{p}^u] \end{aligned} \tag{6.10}$$

Suppose that the deterministic optimization problem of Equation 6.8 obtains n_p number of Pareto-optimum solutions. After robustness evaluation is performed for each Pareto-optimum solution, there will be n_p number of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values obtained from Equation 6.9. These $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values are returned to the deterministic problem in Equation 6.8 and inserted in the set S_f. Likewise, n_p number of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values are inserted in the set S_g. These are defined as: $S_f = \{0, \Delta\mathbf{x}_1^f, \Delta\mathbf{p}_1^f, \ldots, \Delta\mathbf{x}_s^f, \Delta\mathbf{p}_s^f\}$ and $S_g = \{0, \Delta\mathbf{x}_1^g, \Delta\mathbf{p}_1^g, \ldots, \Delta\mathbf{x}_k^g, \Delta\mathbf{p}_k^g\}$, where s and k represent the total number of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values in S_f and S_g, respectively. The improved MORO repeats the first and second step for a number of iterations and the $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values in either S_f or S_g are accumulated, so both s and k can be larger than n_p. In this way, the number of constraint functions defined by $\|[\mathbf{f}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) - \mathbf{f}(\mathbf{x}, \mathbf{p})]^{+}\| \le \eta_f, \forall \Delta\mathbf{x}, \Delta\mathbf{p} \in C_f$ is $m \times (s + 1)$ and the number of constraints defined by $\mathbf{g}(\mathbf{x} + \Delta\mathbf{x}, \mathbf{p} + \Delta\mathbf{p}) \le 0, \forall \Delta\mathbf{x}, \Delta\mathbf{p} \in S_g$ is $n \times (k + 1)$ where m and n represent the number of objective and constraint functions, respectively. Notice that the robustness evaluation in Equation 6.9 and Equation 6.10 is similar to the lower-level subproblems in the nested RMOGA, as defined in Equation 6.6 and Equation 6.7.

The iterative process for the sequential RMOGA is as follows:

- *First iteration*: at the beginning, $S_f = S_g = \{0\}$, which means that in the first step, Equation 6.8 reduces to the original multi-objective optimization problem in Equation 6.1. The Pareto-optimal solutions from Equation 6.8 are obtained. In the second step, the robustness for the Pareto optimum solutions from the first step is evaluated. This robustness evaluation is performed by solving Equation 6.9 and Equation 6.10 for each of the Pareto optimum solutions obtained. Solving each of the maximization problems in Equation 6.9 and Equation 6.10 globally, the optimum value of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ is obtained. This essentially is the worst value of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ considering the variation in the objective and constraint functions. The two pairs of worst values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$, one from Equation 6.9 and the other from Equation 6.10, are inserted in S_f and S_g, respectively. The robustness evaluation is performed for the remaining Pareto optimum solutions one by one. By the end of the second step for all Pareto optimum points in the first iteration, there are an equal number of worst values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ in S_f and S_g. Finally, based on the robustness evaluation—whether the inequality in Equation 6.9 and Equation 6.10 are satisfied, the robust solutions are identified while the nonrobust ones are discarded. This completes a single iteration in the improved MORO.
- *Second iteration:* sequential RMOGA repeats the previous steps in the first iteration except that both S_f and S_g now contain the worst values of $\Delta\mathbf{p}$, from the previous iteration. In this way, the problem in Equation 6.8 has more constraints and becomes more restricted compared to that the one in the first iteration. As a result, the Pareto-optimal solutions from Equation 6.8 may be different from those obtained in the first iteration. Again, the robustness for each Pareto optimum solution obtained in this iteration is evaluated, and additional worst values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ are added to S_f and S_g. The robust Pareto solutions obtained from the second iteration are combined with those from the first iteration, and this completes the second iteration in the sequential RMOGA.
- *Remaining iterations:* the same procedure as in the above iterations is repeated for a number of iterations until the following stopping criteria are satisfied: (i) a maximum number of function calls is reached; (ii) no improvement in the Pareto solutions from one iteration to the next is obtained.

Ideally the number of worst values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ in S_f and S_g should be the same. This is because in the second step, Equation 6.9 and Equation 6.10 are used to evaluate multi-objective and feasibility robustness constraints respectively for the same set of Pareto optimum solutions. Let n_p be the number of Pareto optimum solutions, then the total number of optimum solutions obtained from either Equation 6.9 and Equation 6.10 should be equal to n_p. Although the number of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ in S_f and S_g can be equal, the value of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ obtained from Equation 6.9 is not the same as the values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ from Equation 6.10. Therefore, the set S_f must be different from S_g. Furthermore, some values of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ in either S_f or S_g can be redundant. Therefore, duplicate copies of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values are eliminated by the end of each iteration, and thus the total number of $\Delta\mathbf{x}$, $\Delta\mathbf{p}$ values, as represented respectively by s and k, are also different.

6.3.3 Comparison between Nested and Sequential RMOGA

There are several differences between the nested and sequential RMOGA: (i) by comparing Equation 6.5 with Equation 6.8, it is noted that all constraints in the first-step problem of the sequential RMOGA are explicitly defined as the sample sets $\boldsymbol{\kappa}$, $\boldsymbol{\mu}$ and $\boldsymbol{\lambda}$, $\boldsymbol{\nu}$ are already known. However, this is not the case in the upper level problem in the nested RMOGA where the constraint values (the left-hand side of multi-objective and feasibility robustness constraints, i.e., $\max \|\Delta\mathbf{f}^{+}\|$ and $\max[\max \mathbf{g}]$) must be evaluated by solving two lower-level subproblems. In this way, solving the first-step problem in Equation 6.8 is much more efficient than solving the upper level problem in Equation 6.5. (ii) In the nested RMOGA, the upper level problem in Equation 6.5 is solved by MOGA only once during the entire procedure. Because the upper level problem evaluates multi-objective and feasibility robustness constraints for all intermediate points in the population, the solutions from upper level problems include only robust solutions. In the sequential RMOGA, the first-step problem in Equation 6.8 is solved by MOGA and produces some candidate optimal solutions. Because the first-step problem does not evaluate multi-objective and feasibility robustness constraints, the solutions from the first-step problem are not necessarily robust solutions. Therefore, the second-step problems are used to evaluate robustness for the candidate optimal solutions obtained from the first-step problem. (iii) In the nested RMOGA, the lower level subproblems defined in Equation 6.6 and Equation 6.7 does not evaluate multi-objective and feasibility robustness constraints; they just provide $\max \|\Delta\mathbf{f}^{+}\|$ and $\max\,[\max \mathbf{g}]$ values for the upper-level problem. On the other hand, the second-step problems in Equation 6.9 and Equation 6.10 in the sequential RMOGA evaluate multi-objective and feasibility robustness constraints. In addition, the second-step problems provide the optimal solutions—$\Delta\tilde{\mathbf{x}}$ and $\Delta\tilde{\mathbf{p}}$ as sample values to the first-step problem.

Because of the differences between the two RMOGA approaches as listed above, the sequential approach requires considerably less computational effort and can be more efficient than the nested approach. On the other hand, and in general, the sequential RMOGA may not be able to obtain all robust solutions that can be obtained by the nested RMOGA. One can compare the required number of function calls by the nested and sequential RMOGA as follows: suppose all optimization problems in both nested and sequential RMOGA use the same number of generations and population size for MOGA and GA. Let n_g be the generation and n_{ps} be the population size. In the sequential RMOGA, let n_t represents the number of iterations and n_s represents the average number of optimal solutions obtained from the first-step subproblems. Then the total number of function calls for the nested RMOGA is of the order $\Theta(n_g^2 \times n_{ps}^2)$ while the total number of function calls in the sequential RMOGA is of the order $\Theta[n_t \times (n_g \times n_{ps} + n_s \times n_g \times n_{ps})]$ which has the same order of magnitude as $\Theta[n_t \times n_s \times n_g \times n_{ps})]$. The number of iterations in the sequential RMOGA is much smaller than the number of generations of GA, for example as shown in the case-study section, the iteration of sequential RMOGA is 5 while the GA generation is 50. The average number of optimal solutions must also be smaller than the population size. Since $n_t < n_g$ and $n_s < n_{ps}$, it follows that $\Theta(n_t \times n_s \times n_g + n_{ps}) < \Theta(n_g^2 \times n_{ps}^2)$. Therefore, the number of function calls by the sequential RMOGA can be significantly less than that by the nested RMOGA.

Nevertheless, both the nested and sequential RMOGA can become computationally expensive. To reduce their computation effort, an online approximation approach is developed and combined with both the nested and sequential RMOGA, as described next.

6.4 Online Approximation-Assisted RMOGA

In general, an approximation technique consists of three main steps: sampling, metamodeling and verification. The sample space is a multi-dimensional space in which the coordinate axes represent the variables of an optimization. In sampling, a number of points are selected from the sample space and then their objective and constraint function values are obtained (or "observed"). Approximation can be done both offline or online. For example, offline sample points can be selected based on space-filling criteria (Koehler, 1996) and done before the optimization process is initiated; while online sample points are determined adaptively in order to satisfy certain goals. In RMOGA one important goal is to locate and observe a limited number of sample points while satisfying the accuracy for all objective and constraint functions. Typically, an online sampling technique requires fewer number of sample points than an offline technique to achieve the same accuracy. Based on the sample points, the next step is metamodeling, which constructs metamodels to approximate the objective and constraint functions. Many metamodeling techniques have been successfully applied in engineering optimization problems (Wang and Shan, 2007), for example Kriging, radial basis function, neural network, multivariate adaptive regression splines etc. Specially, Kriging is a Gaussian interpolation method that is reliable and consistent in building accurate metamodels (Sacks *et al.*, 1989). Using the metamodels of the objective and constrain functions, the approximation-assisted optimization is able to obtain an estimate of the optimum solutions. The final step in approximation is to verify the accuracy of these estimated solutions. In the following, the steps in online approximation assisted RMOGA approaches are first presented. The sampling in online approximation in discussed in section 6.4.2 and metamodeling and verification are presented in section 6.4.3. Finally, in section 6.4.4, the sample selection and filtering strategy in online approximation in RMOGA is provided.

6.4.1 Steps in Approximation-Assisted RMOGA

The flowcharts for approximation assisted RMOGA approaches are illustrated with the flow diagram in Figure 6.3(a) for the nested approach and in Figure 6.3(b) for the sequential approach.

For the nested RMOGA, Figure 6.3(a), initially the metamodels for the objective and constraint functions are generated in the "Offline approximation" block based on a space-filling (offline) technique. The metamodels are forwarded to the "Nested RMOGA" block in which upper- and lower-level problems, as defined in Equation 6.5 and Equation 6.6 are solved. When the nested RMOGA obtains an estimated set of optimum solutions, it sends them to the 'Online approximation' block, in which the online sample points are determined (discussed in section 6.4.2). Next, the online samples are combined with the previous (offline) samples and used to update the metamodels. While the stopping criteria (discussed in section 6.3.1) are not satisfied, the updated metamodels are returned to the 'Nested RMOGA' block and the previous steps are repeated. For the sequential RMOGA,

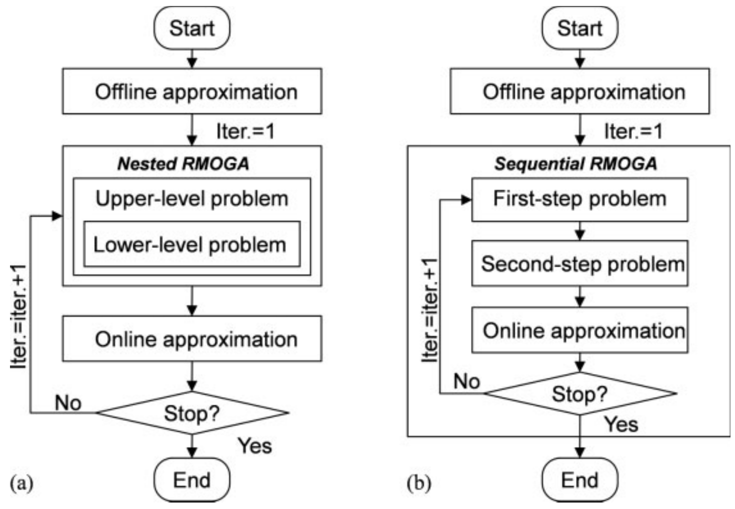

Figure 6.3 *Flow diagram for online approximation assisted RMOGA: (a) nested approach, (b) sequential approach.*

as in Figure 6.3(b), it also starts with the "offline approximation" block. However, because the sequential RMOGA needs to iterate between its first-step and second-step problems, the "online approximation" step is performed at the end of each iteration of sequential RMOGA as shown in Figure 6.3(b). Notice that this is different in the nested RMOGA where online approximation is performed after nested RMOGA obtains the final estimated optimum solutions.

One comment in online approximation is that the sample space in RMOGA is a combination of both variables and uncertain parameters space. To determine the values of variables for each online sample point, the approximation technique uses the optimum value of variables from each optimal solution. The values of uncertain parameters for the sample points are determined in two different ways in the nested and sequential RMOGA. In the nested RMOGA, samples for the parameters are generated in a Latin Hypercube around each nominal point. However in the sequential RMOGA, the first-step problem determines the optimum values of variables and the second-step problem obtains the optimum values for the parameters. The optimum value for parameters from the second step are paired with the optimum value of variables and used for online sampling.

6.4.2 Sampling

The online approximation in both RMOGA approaches is based on a two-stage sampling strategy which combines both offline and online sampling. The first-stage samples are placed offline in the entire sample space using a space-filling sampling method before RMOGA starts. The second-stage samples are placed online based on the optimum solutions generated by RMOGA. The detail of the two-stage sampling strategy is presented below.

The offline samples are generated initially based on a commonly used space-filling sampling technique called Latin Hypercube Sampling (LHS) (Koehler and Owen, 1996).

These sample points are used to construct a metamodel for each objective and constraint function required by RMOGA. Note that each sample point needs to be observed once for all functions. Using the metamodels of the objective and constraint functions, RMOGA obtains a set of estimated optimal solutions. From these estimated optimal solutions, a few solution points are selected (the selection scheme is presented in section 6.4.4) and observed, which are designated as the online samples. Both online and (previously obtained) offline samples are combined and used to reconstruct/update the metamodels for the objective and constraint functions. Once the metamodels are updated, online sampling is repeated until RMOGA progressively approaches the true optimum solutions.

One motivation to use the estimated optimum solution for online sampling is that they are potentially located close to the true optimum solutions. By observing the online sample points, the accuracy for all objective and constraint functions in the nearby region can be expected to be significantly improved. This will be beneficial for RMOGA to obtain a good estimate of the Pareto optimum solutions.

6.4.3 Metamodeling and Verification

In the online approximation-assisted RMOGA, Kriging is used as the metamodeling technique for all the objective and constraint functions. Let $y(\mathbf{x})$ represents an objective or constraint function that should be approximated; the Kriging model is presented as:

$$y(\mathbf{x}) = \delta + Z(\mathbf{x}) \tag{6.11}$$

where $\mathbf{x}$ is a point (vector) in the sample space, δ is a known global function and $Z(\mathbf{x})$ is the realization of a Gaussian random process with mean zero and variance σ^2 (Koehler and Owen, 1996). While δ captures the overall trend of the Kriging model, $Z(\mathbf{x})$ is used to represent a local deviation from the global function. Typically, the sample points are interpolated with the Gaussian random function to estimate the stochastic process. Given a total number of n sample points, the covariance of $Z(\mathbf{x})$ at two sample points $\mathbf{x}^i$ and $\mathbf{x}^j$ can be expressed as in Equation 6.12:

$$Cov[Z(\mathbf{x}^i), Z(\mathbf{x}^j)] = \sigma^2 R(\mathbf{x}^i, \mathbf{x}^j) \tag{6.12}$$

where $R(\mathbf{x}^i, \mathbf{x}^j)$ is the Gaussian correlation function between sample points $\mathbf{x}^i$ and $\mathbf{x}^j$. The Kriging predictor is given in Equation 6.13:

$$\hat{y}(\mathbf{x}) = \hat{\delta} + \mathbf{r}'\mathbf{R}^{-1}(\mathbf{y} - \mathbf{1}\hat{\delta}) \tag{6.13}$$

where $\hat{y}$ is the predicted (metamodel) value (predictor) of y and $\hat{\delta}$ is the predicted value of δ, which is the expected value of the posterior process. $\mathbf{R}$ is a $n \times n$ matrix whose (i, j) element is $\mathrm{Cov}[Z(\mathbf{x}^i), Z(\mathbf{x}^j)]$, $\mathbf{r}$ (with prime superscript for transpose) is the vector whose ith element expressed as:

$$r_i(\mathbf{x}) = Cov[Z(\mathbf{x}), Z(\mathbf{x}^i)] \tag{6.14}$$

Based on the Kriging model, an estimated mean square error (mse) at an unobserved point is given in Equation 4.5:

$$mse(\mathbf{x}) = \sigma^2(1 + c\mathbf{R}c' - 2cr) \tag{6.15}$$

where c is a vector of Kriging coefficients. Based on Equation 6.15, the Kriging mse accounts for the correlations between an unobserved point and the sample (observed) points. Since the correlation between points decreases as the distance between them increases, an unobserved point with a large Kriging *mse* indicates a poor correlation with the existing sample points. In this way, the predicted function values at such a point can be inaccurate. In the online approximation assisted RMOGA approaches, the *mse* at the estimated optimal solutions are calculated using Equation 6.15. In case that the error is larger than a user-specified tolerance value, additional sample points should be considered and observed in order to increase the accuracy of the metamodel.

6.4.4 Sample Selection and Filtering

According to the steps of the approximation assisted RMOGA, as in section 6.4.1, a large number of (intermediate) estimated optimum solutions are generated from the online approximation. It is impractical to observe all the estimated optimum solutions, so it is necessary to prioritize the optimum solutions and only observe a subset among them. One way to rank the optimum solutions is by using the *mse*, which can be calculated from Equation 6.15. From a sampling point of view, it is more desirable to observe a sample with relatively larger *mse* in order to improve the overall accuracy of the approximation.

On the other hand, as multiple functions (objective and constraint) need to be approximated through Kriging at each unobserved point, it is important to define a scalar as a measure of the overall accuracy for the estimated functions as shown in Equation 6.16:

$$error = \left(\sum mse_f^2 + \sum mse_g^2\right)^{1/2} \tag{6.16}$$

where mse_f and mse_g represent the Kriging *mse*'s of the objective and constraint functions respectively. It should be mentioned that the values of the objective and constraint functions must be normalized so their Kriging *mse* calculated from Equation 6.15 are in the same scale—for example, the values of objective and constraint functions can be normalized using the largest absolute values for corresponding functions. Based on the calculation from Equation 6.16, an estimated optimum solution with the largest error is ranked first followed by the one with the second largest error, and so on. Based on the ranking of the optimum solutions, one can select (using the error-based ranking) one or a number of solution points to observe during the online approximation in RMOGA.

In addition to the scheme in selecting the estimated optimum solutions, a sample filtering is also included in the online approximation to prevent clustering of sampled points. When the distance (measured in the sample space) between a new sample and a previous sample point is less than a threshold value, the new sample point is eliminated. In approximation assisted RMOGA, the following filtering criterion can be used: $\|\mathbf{x}_n - \mathbf{x}_e\| \geq \varepsilon$, where $\mathbf{x}_n$ refers to a new sample point and $\mathbf{x}_e$ refers to any existing sample points. $\|\cdot\|$ denotes the Euclidean norm (distance) between two vectors in the sample space, and ε is a user defined threshold value specifying the minimum acceptable distance between two sample points. As a general rule, the threshold is selected such that it is at least larger than half of the shortest distance among existing sample points (measured pairwise). The computational costs of the optimization models must also be considered in selecting the threshold ε. When the objectives and constraint functions are computationally expensive to compute, the value of ε needs to be increased in order to reduce the number of online samples. After additional sample points are determined and the actual simulations are evaluated to obtain

Table 6.1 *Genetic algorithm parameter settings.*

Parameter	Upper level/first step	Lower level/second step
Population size	$15i$	$15j$
Maximum generation	50	50
Elite number[a]	$1(i < 5)2(i > 5)$	1
Crossover probability	0.9	0.9
Mutation probability	0.1	0.1

[a] *i* and *j* are the number of variables and uncertain parameters, respectively

the response values, these sample points are added to the current set of sample points. Finally, the updated sample points are used to update the metamodels.

6.5 Case Studies

In this section the results for the application of RMOGA to one numerical and one engineering example are presented. For both nested and sequential RMOGA, the maximum number of iterations is set equal to 5 to allow a sufficient number of robust solutions is obtained. The multi-objective genetic algorithm of MATLAB$^{\text{TM}}$ "Global Optimization Toolbox" version 2010a (Mathwork, 2010) is used as the optimizer. Parameter settings for the genetic algorithm in all examples are shown in Table 6.1.

Kriging (Koehler and Owen, 1996) is used for constructing metamodels in all examples, where a second order polynomial function is selected to build the regression model and a Gaussian function is used for the correlation model. The initial offline samples are generated using a Latin Hypercube Sampling (LHS) (Koehler and Owen, 1996) technique with $(N+1)(N+2)/2$ number of samples (minimum requirement), where N is the total number of variables and uncertain parameters.

6.5.1 Numerical Example

The first example, well known in the literature as 'TNK', is a numerical bi-objective optimization problem adapted from a previous work (Deb, 2001). The problem formulation is shown in Equation 5.1:

$$\begin{aligned}
\min \quad & f_1 = x_1 \\
\min \quad & f_2 = x_2 \\
\text{s.t} \quad & g_1 = 1 + 0.1\cos\left(16\arctan\frac{x_1}{x_2}\right) + 0.2\sin(p_1)\cos(p_2) - x_1^2 - x_2^2 \leq 0 \\
& g_2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \leq 0.5 \\
& 0 < x_1, x_2 \leq \pi
\end{aligned} \tag{6.17}$$

This optimization problem has two design variables and two uncertain parameters. The nominal values for both parameters are 1, i.e., $p_1 = p_2 = 1$. The feasible domain is defined by the non-convex area within two inequality constraints g_1 and g_2. The optimum solutions to the problem in Equation 6.17 are obtained using a deterministic MOGA and shown as in

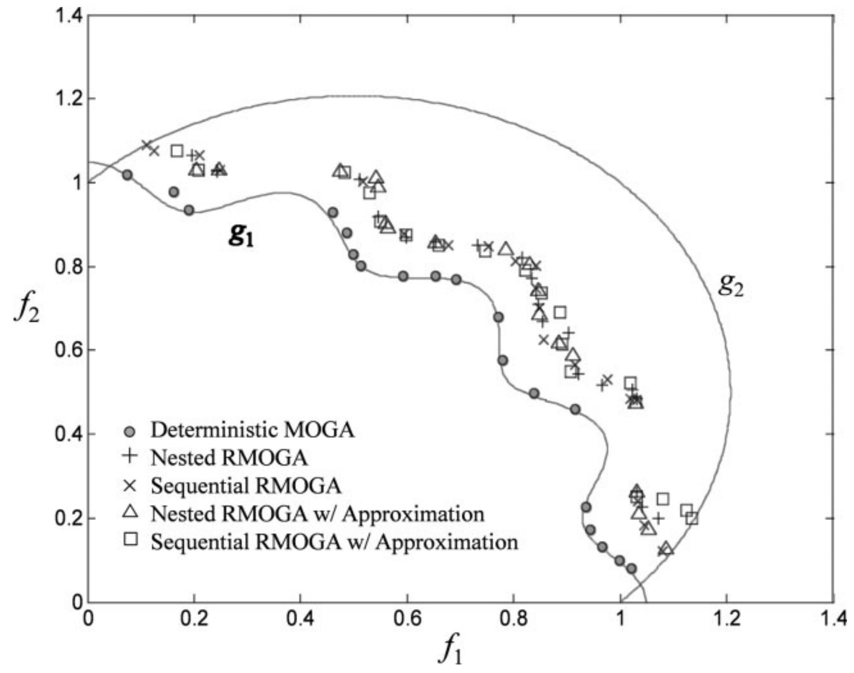

Figure 6.4 *Optimum solutions in numerical example (TNK).*

Figure 6.4. Notice that these solutions are located along the boundary of constraint g_1. Due to the non-convexity of g_1, as shown in Figure 6.4, the Pareto frontier of the deterministic MOGA solutions consists of three discontinuous sections.

By considering interval uncertainty in the parameters, the optimization problem in Equation 6.17 can be formulated using the nested RMOGA as an upper level problem and a lower-level sub-problem in Equation 6.18 and Equation 6.19 respectively:

$$\begin{aligned}
\text{min}\quad & f_1 = x_1 \\
\text{min}\quad & f_2 = x_2 \\
\text{s.t}\quad & g_1 = 1 + 0.1\ \cos\left(16\ \arctan\frac{x_1}{x_2}\right) + 0.2\ \sin(p_1)\cos(p_2) - x_1^2 - x_2^2 \le 0 \\
& g_2 = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 \le 0.5 \\
& \max_{\Delta\mathbf{p}}[\max(g_1, g_2)] \le 0 \\
& 0 < x_1, x_2 \le \pi
\end{aligned} \tag{6.18}$$

$$\begin{aligned}
\max_{\Delta\mathbf{p}}[\max(g_1, g_2)] = \max_{\Delta\mathbf{p}}\ & 1 + 0.1\cos\left(16\ \arctan\frac{x_1}{x_2}\right) \\
& + 0.2\sin(p_1 + \Delta p_1)\cos(p_2 + \Delta p_2) - x_1^2 - x_2^2 \\
\text{s.t}\ \Delta p_1, \Delta p_2 \in [-2, 2] &
\end{aligned} \tag{6.19}$$

where Δp_1 and Δp_2 represent the uncertainty in the parameters and their uncertainty ranges are both between ± 2, as specified in Equation 6.18. Notice that the upper level problem in Equation 6.18 reduces to the original formulation in Equation 6.17 when $\Delta p_1 = \Delta p_2 = 0$.

Because there is no uncertainty in the objective functions and constraint g_2, only feasibility robustness for constraint g_1 is considered in this example. The lower level subproblem in Equation 6.19 is essentially a single-objective maximization of g_1. On the other hand, the optimization problem in Equation 6.17 can also be formulated using the sequential RMOGA as first- and second-step problems in Equation 6.20 and Equation 6.21 respectively:

$$\begin{aligned}
\min\quad & f_1 = x_1 \\
\min\quad & f_2 = x_2 \\
\text{s.t.}\quad & g_1 = 1 + 0.1\cos\left(16\arctan\frac{x_1}{x_2}\right) + 0.2\sin(p_1+\Delta p_1)\cos(p_2+\Delta p_2) \\
& \qquad - x_1^2 - x_2^2 \le 0, \forall \Delta p_1, \Delta p_2 \in S_g \\
& g_2 = (x_1-0.5)^2 + (x_2-0.5)^2 \le 0.5 \\
& 0 < x_1, x_2 \le \pi
\end{aligned} \tag{6.20}$$

$$\begin{aligned}
\max_{\Delta \mathbf{p}}\ & 1 + 0.1\cos\left(16\arctan\frac{x_1}{x_2}\right) + 0.2\sin(p_1+\Delta p_1)\cos(p_2+\Delta p_2) \\
& \qquad - x_1^2 - x_2^2 \le 0 \\
\text{s.t.}\ & \Delta p_1, \Delta p_2 \in [-2, 2]
\end{aligned} \tag{6.21}$$

where S_g represents a set of Δp_1 and Δp_2 values in the uncertain interval. As mentioned earlier, S_g is determined after robustness evaluation in Equation 6.21. Next, the optimization problem defined in Equation 6.18 and Equation 6.19 (nested formulation), and that in Equation 6.20 and Equation 6.21 (sequential formulation) are solved. For comparison, the nested and sequential RMOGA approaches are each applied for the numerical example with and without using the online approximation technique. When online approximation is used, the metamodels are developed only for the constraint functions but not for the objective functions due to the simplicity of the objective functions. The number of initial (offline) sample points is 15. To account for the randomness in GA, all RMOGA approaches were repeatedly run ten times, among which the best solutions are selected and plotted in Figure 6.4. Notice that the 'Nested RMOGA' and 'Sequential RMOGA' in Figure 6.4 refer to the approaches without using online approximation.

According to Figure 6.4, the optimal solutions from both RMOGA approaches are inferior to the deterministic solutions, which are expected because the robust solutions are typically more conservative than the deterministic ones. On the other hand, the optimal solutions from both nested and sequential RMOGA approaches, either with or without using the online approximation are generally consistent in the objective space. Table 6.2 and

Table 6.2 *Comparison between nested and sequential RMOGA for the numerical example.*

	Nested RMOGA		Sequential RMOGA	
	Mean	Std.	Mean	Std.
Num. function evaluations	1,897,820	0	11,411	1164
Hyperarea difference (HD)	0.602	0.04	0.534	0.01
Overall Pareto spread (OS)	0.272	0.09	0.445	0.03

Table 6.3 *Comparison between approximation-assisted RMOGA approaches for the numerical example.*

	Nested RMOGA		Sequential RMOGA	
	Mean	Std.	Mean	Std.
Num. online samples	95	7.6	26	5.4
Hyperarea difference (HD)	0.587	0.03	0.581	0.04
Overall Pareto spread (OS)	0.392	0.03	0.340	0.08

Table 6.3 compared the obtained mean value and standard deviation information for different RMOGA approaches. When online approximation technique is not used (Table 6.2), the nested RMOGA requires a large number of function calls; while the sequential RMOGA requires considerably (about two orders of magnitude) less function calls. The quality metrics (Wu and Azarm, 2001): Hyperarea Difference (HD) and Overall Pareto Spread (OS) are calculated to measure the goodness of the Pareto solutions. It is found that the optimal solutions obtained from the sequential RMOGA are slightly better than the solutions obtained from the nested RMOGA because the lower level subproblem in the nested RMOGA makes it more restricted.

For the RMOGA approaches with online approximation (Table 6.3), the number of sample points can be taken as the counterpart of the number of function calls in RMOGA approaches without using online approximation. It is observed that both approximation-assisted RMOGA approaches significantly reduce the number of calls for the evaluation of the constraint functions. On average, the nested RMOGA converges in three iterations with 95 online sample points (excluding the initial 15 samples); while the sequential RMOGA requires far fewer sample points but needs more iterations to converge. The maximum root mean square errors calculated from the Kriging model (recall Equation 6.15) for the final optimal solution in both approximation-assisted RMOGA approaches are reasonably small. Based on the quality metrics calculated in Table 6.3, it is again observed that the optimal solutions from the sequential approach are slightly better than the solutions from the nested approach in online approximation assisted RMOGA. A separate validation using Monte Carlo simulation on all the final solutions was also performed to ensure feasibility robustness for the optimal solutions are satisfied. Finally, the mean squared errors for optimal solutions from both approximation assisted RMOGA approaches are computed, which are found to be less than the acceptable value (0.01).

6.5.2 Oil Refinery Case Study

In the second case study, a typical crude oil refinery is considered, and the nested and sequential RMOGA approaches are employed for optimization. The refinery consists of common unit process/operations, and nonlinear correlations are used to predict the yields and properties of the products of each unit. The units in this refinery case study are: (i) crude distillation unit; (ii) delayed coker; (iii) hydrocracker for heavy vacuum gas oils; (iv) hydrotreater for light vacuum gas oils; (v) fluid catalytic cracking unit (FCCU); (vi) hydrotreater for heavy straight run naphtha: (vii) catalytic reformer; (viii) light naphtha hydrotreater; (ix) isomerization unit.

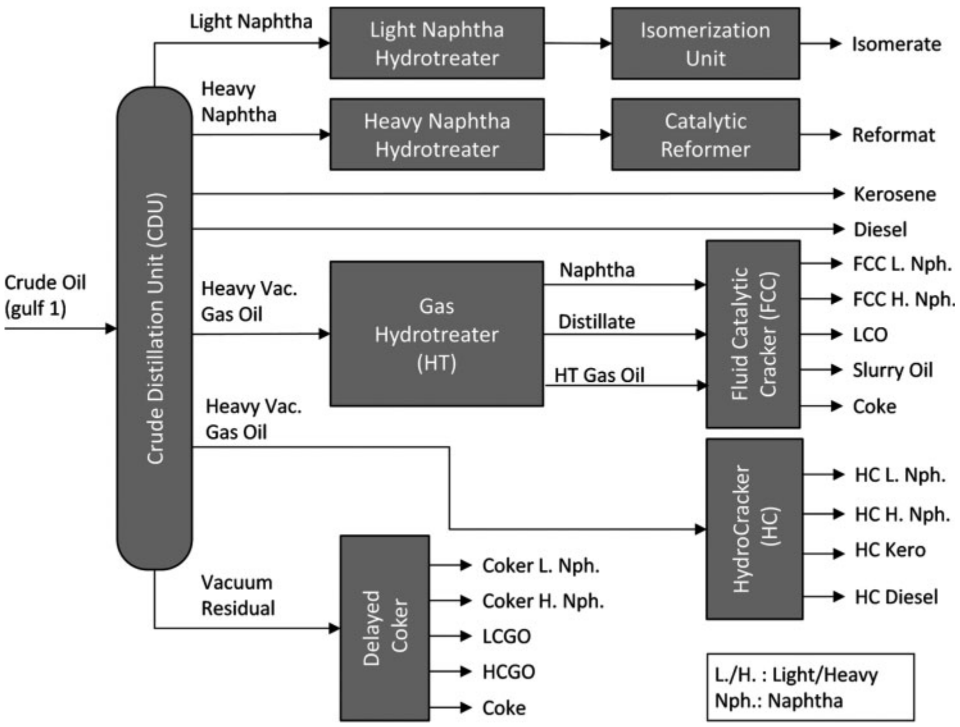

Figure 6.5 *Schematic of refinery model.*

The schematic of the oil refinery is shown in Figure 6.5. The flow diagram depicts various unit processes and flows of intermediate product streams. The products from the crude distillation unit are lower straight run (LSR) naphtha, higher straight run (HSR) naphtha, straight run diesel (SRD), kerosene, light vacuum gas oil (LVGO), heavy vacuum gas oil (HVGO) and vacuum residue (VacResid). The vacuum residue is further processed in the delayed coker to get the lighter fractions. The heavy vacuum gas oils are hydrocracked in the hydrocracker to get light naphtha and heavy naphtha fractions. The LVGO, HSR and LSR are hydrotreated to reduce the sulfur contents and further treated in FCCU, catalytic reformer and isomerization unit respectively to get the products of interest. All naphtha is sent to the blending pool to get the gasoline for the required grade.

The simulation of the described refinery is done through MATLAB and simple nonlinear correlations are used. The flow rate of crude oil to the crude distillation unit is assumed to be fixed with a value of 100 000 BPD. For simplicity, the schematic in Figure 6.5 does not include the utility units such as steam, cooling water and electricity. The storage facilities such as crude oil and intermediate product storage tanks are also not shown.

The refinery model is formulated as a MOO problem as described in Equation 6.22. The two objectives are to maximize the product flow rate f_1 and to minimize the cost f_2. Both objectives can be evaluated from the refinery simulation model for a given set of design variables. The variables considered for optimization are the six cut temperatures ($t_i, i = 1, \ldots, 6$) in the crude distillation unit. The lower and upper bounds for the cut temperatures are given in Equation 6.22. It is assumed that t_2 and t_3 are uncertain and the uncertainties are represented by Δt_i, $j = 1, 2$, and the range of uncertainties are between $\pm$ 10% of their nominal cut temperature values.

$$\begin{aligned} \max \quad & f_1(t_i) = \text{flow rate of light naphtha(bb1/day)} \\ \min \quad & f_2(t_i) = \text{total cost(\$/day)} \\ \text{s.t.} \quad & \left\| [f_m(t_i, \Delta t_j) - f_m(t_i)]^+ \right\| \le \eta_f \\ & \forall \Delta t_j^l \le \Delta t_j \le \Delta t_j^u, \; j = 1, 2, m = 1, 2 \\ & 162 \le t_1 \le 198 \\ & 360 \le t_2 \le 440 \\ & 477 \le t_3 \le 583 \\ & 585 \le t_4 \le 715 \\ & 810 \le t_5 \le 950 \\ & 950 \le t_6 \le 1155 \end{aligned} \tag{6.22}$$

In the refinery example, the absolute value for the two objective functions are not on the same scale—for example, the flow rate of light naphtha and total cost are in the order of 10^4 and 10^6 respectively. As such, the original value of the flow rate and total cost are first normalized to a value of unity using normalizing factors 10^5 and 10^7, respectively. The advantage of normalization is that using a Euclidean norm to restrict the objective variation as shown by the inequality constraint in Equation 6.22, will give equal importance for both objectives. The acceptable variation limit η_f is specified as 0.1 in the refinery example. The optimization problem given in Equation 6.22 is solved using both the nested and the sequential RMOGA approaches. For comparison purposes, the deterministic optimal solutions (assuming no uncertainty in the input) are also obtained. Both RMOGA approaches are run for a total of ten times and a best set of optimal solutions out of the ten runs are selected for each approach. The best sets of optimal solutions for both RMOGA approaches are plotted in the objective functions space in Figure 6.6. It can be seen that the optimal solutions for the refinery example from both RMOGA approaches are consistent as well. It is also observed that the deterministic optimal solutions are better than both RMOGA approaches in achieving maximum flow rate of light naphtha, while the uncertainty in the cut temperature seems to have little effect on the daily total cost.

The average value and standard deviation of the optimal solutions based on the ten runs for each of the RMOGA approach are shown in Table 6.4. The average number of iterations for both RMOGA approaches in the refinery example is three. In terms of the quality of the obtained Pareto frontier, the sequential RMOGA performs slightly better (with a higher

Table 6.4 *Comparison between nested and sequential RMOGA for the refinery example.*

	Nested RMOGA		Sequential RMOGA	
	Mean	Std.	Mean	Std.
Hyperarea difference (HD)	0.676	0.02	0.660	0.01
Overall Pareto spread (OS)	0.177	0.07	0.219	0.05

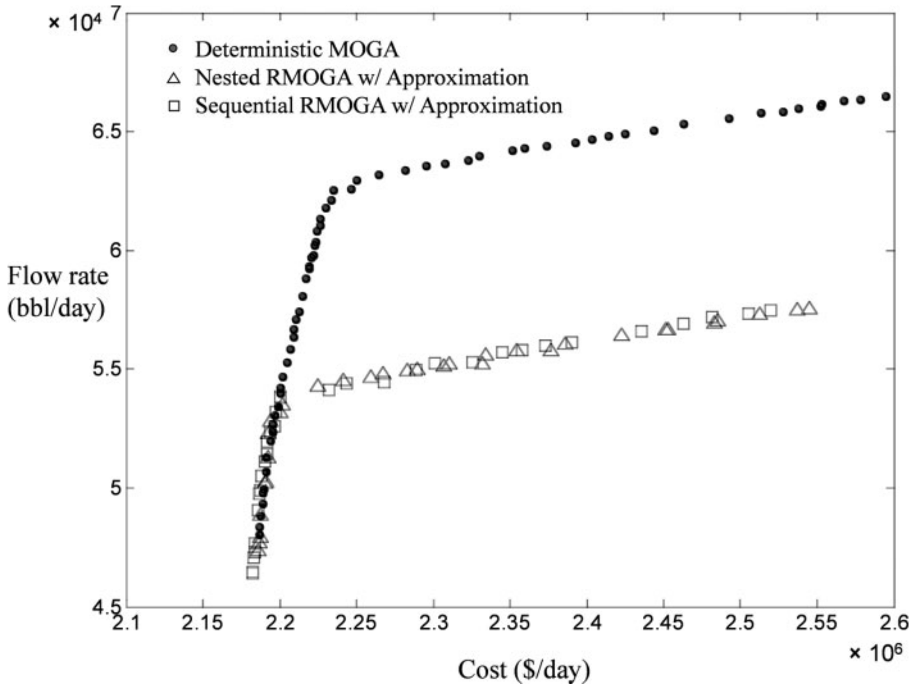

Figure 6.6 *The optimal solutions for the refinery example.*

spread) than the nested approach. The number of total samples in both approaches is 28. Based on the mean squared error in the approximated objective functions (less than 0.001), Kriging provides good accuracy in both RMOGA approaches. This is possibly due to the good characterization of the polynomial relationship between the input and output variables in the refinery model.

6.6 Conclusions

In this chapter, two RMOGA approaches were presented to address the challenge of optimization under uncertainty and the associated computational cost. Using intervals to represent the uncertainties in both variables and parameters, RMOGA evaluates the robustness of solutions in terms of their objective and constraint functions with a worst-case analysis. It was shown that a sequentially formulated RMOGA can be more efficient than a nested RMOGA approach. However, the computational cost for applying RMOGA on many chemical engineering applications can be intractable. To overcome this difficulty, the online approximation method was integrated with RMOGA to replace a potentially expensive function with an inexpensive metamodel or surrogate. A specially developed online sampling technique selects and observes the optimum solutions, and then uses them as additional samples to improve the accuracy of approximation. This potentially improves the accuracy of approximated objective and constraint function values at the optimum region. Two examples are used to shown the applicability of the RMOGA approaches. Both approaches in RMOGA are able to obtain a good estimate of the optimum solution while retaining a reasonably small amount of function calls. In the numerical example, the optimum solutions from the RMOGA approaches are compared with and without using online approximation. It is found that using online approximation can significantly reduce the number of function calls in both RMOGA approaches. The oil refinery case study shows the applicability of RMOGA in simulation based optimization problems in which the optimum solutions from both RMOGA approaches are consistent.

Acknowledgments

The material in this chapter was the result of a research collaboration between the Petroleum Institute and University of Maryland, College Park, as part of the Education and Energy Research Collaboration (EERC) agreement. Such support does not constitute an endorsement by the funding agency of the opinions expressed in this chapter.

References

Al-Sharrah, G.K., Alatiqi, I., Elkamel, A., and Alper, E. (2001). Planning an integrated petrochemical industry with an environmental objective. *Industrial and Engineering Chemistry Research*, *40*(*9*), 2103–2111.

Basseur, M., and Zitzler, E. (2006). A preliminary study on handling uncertainty in indicator-based multiobjective optimization, *Applications of Evolutionary Computing*, Springer, Berlin, pp. 727–739.

Coello Coello, C.A. and Lechuga, M.S. (2002). MOPSO: A proposal for multiple objective particle swarm optimization. *Congress on Evolutionary Computation (CEC'2002)*, *2*, 1051–1056.

Coello Coello, C.A. (2000). Multiobjective optimization of trusses using genetic algorithms. *Computers and Structures*, *75*(*6*), 647–660.

Coello Coello, C.A., Veldhuizen, D.A. and Lamont, G.B. (2002). *Evolutionary Algorithms for Solving Multi-Objective Problems*, second edition, Springer, Berlin.

Cohon, J.L. (1978). *Multiobjective Programming and Planning*, Academic Press, New York.

Deb, K. (2001). *Multi-Objective Optimization Using Evolutionary Algorithms*, Wiley, New York.

Deb, K. (2007). Current trends in evolutionary multi-objective optimization. *International Journal for Simulation and Multidisciplinary Design Optimization*, *8*(*1*), 1–8.

Deb, K. and Gupta, H. (2005). *Searching for Robust Pareto-Optimal Solutions in Multi-Objective Optimization. Evolutionary Multi-Criterion Optimization. EMO 2005*, Springer, Berlin, pp. 150–164.

Ehrgott, M. (2005). *Multicriteria Optimization*, second edition. Springer, Berlin.

Evans, G.W., Moreno, J.A. and Mollaghasemi, M. (1991). Evolutionary multi-objective optimization of simulation models. *IEEE 1991 Winter Simulation Conference Proceedings*, pp. 242–250.

Ferreira, J., Fonseca, C.M. and Covas, J.A. (2008). Evolutionary multi-objective robust optimization. *Advances in Evolutionary Algorithm*, I-Tech Education and Publishing, Vienna, pp. 261–278.

Fonseca, C.M. and Fleming, P.J. (1993). Genetic algorithms for multiobjective optimization: formulation, discussion and generalization. *Proceedings of the Fifth International Conference on Genetic Algorithms*, pp. 416–423.

Goldberg, D.E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Boston, MA.

Gunawan, S. and Azarm, S. (2005). Multi-objective robust optimization using a sensitivity region concept. *Structrual and Multidisciplinary Optimization, 29*(*1*), 50–60.

Halsallwhitney, H. and Thibault, J. (2006). Multi-objective optimization for chemical processes and controller design: approximating and classifying the Pareto domain. *Computers and Chemical Engineering*, *30*(*6–7*), 1155–1168.

Holland, J. (1975). *Adaptation in Natural and Artificial Systems*, University of Michigan Press, Ann Arbor.

Hu, W., Li, M., Azarm, S. and S. Almansoori, A. (2011). Multi-objective robust optimization under interval uncertainty using online approximation and constraint cuts. *Journal of Mechanical Design*, *133*(*6*), p. 061002 (9 pages).

Ignizio, J.P. (1978). A review of goal programming: a tool for multiobjective analysis. *Journal of the Operational Research Society*, *29*(*11*), 1109–1119.

Kennedy, J. and Eberhart, R.C. (1995). Particle swarm optimization. *Proceedings of the 1995 IEEE International Conference on Neural Networks*, 1942–1948.

Kirkpatrick, S., Gelatt, C.D. and Vecchi, M.P. (1983). Optimization by simulated annealing, *Science*, *220*, 671–680.

Knowles, J. and Corne, D. (1999). The Pareto archived evolution strategy: a new baseline algorithm for Pareto multiobjective optimisation. *IEEE Proceedings of the 1999 Congress on Evolutionary Computation*, 98–105.

Koehler, J.R. and Owen, A.B. (1996). Computer experiments. *Handbook of Statistics*, Elsevier Science, Amsterdam.

Kurpati, A., Azarm, S. and Wu, J. (2002). Constraint handling improvements for multi-objective genetic algorithms. *Structural and Multidisciplinary Optimization*, *23*(*3*), 204–213.

Li, M., Azarm, S. and Boyars, A. (2006). A new deterministic approach using sensitivity region measures for multiobjective robust and feasibility robust design optimization. *Journal of Mechanical Design*, *128*(*4*), 874–883.

Limbourg, P. (2005). Multi-objective optimization of problems with epistemic uncertainty. *Evolutionary Multi-Criterion Optimization*, Springer, Berlin, pp. 413–427.

Mathwork (2010). *Matlab,* Version 2010a, http://www.mathworks.com (accessed 1 December, 2012).

Miettinen, K. (1999). *Nonlinear Multi-Objective Optimization*, Kluwer Academic Publishers, Boston.

Mitra, K. and Majumder, S. (2011). Successive approximate model based multi-objective optimization for an industrial straight grate iron ore induration process using evolutionary algorithm. *Chemical Engineering Science*, *66*(*15*), 3471–3481.

Nikolaidis, E., Chen, S., Cudney, H., Haftka, R.T. and Rosca, R. (2004). Comparison of probability and possibility for design against catastrophic failure under uncertainty. *Journal of Mechanical Design*, *126*(*3*), 386–394.

Qu, B.Y. and Suganthan, P.N. (2011). Constrained multi-objective optimization algorithm with an ensemble of constraint handling methods. *Engineering Optimization*, *43*(*4*), 403–416.

Rangaian, G.P. (2009). Multi-objective optimization applications in chemical engineering. *Multi-Objective Optimization: Techniques and Applications in Chemical Engineering*, World Scientific, Singapore, pp. 27–54.

Ray, T., Isaacs, A. and Smith, W. (2009). Surrogate assisted evolutionary algorithm for multi-objective optimization. *Multi-Objective Optimization: Techniques and Applications in Chemical Engineering*, World Scientific, Singapore, pp. 131–149.

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

Serafini, P. (1992). Simulated annealing for multiple objective optimization problems. *Proceedings of the Tenth International Conference on Multiple Criteria Decision Making*, 87–96.

Taguchi, G., (1987). *Systems of Experimental Design*, Vols. 1 and 2, Kraus International, New York.

Voutchkov, I. and Keane, A. (2010). Multi-objective optimization using surrogates. *Computational Intelligence in Optimization*, Springer, Berlin, pp. 155–175.

Wang, G. and Shan, S. (2007). Review of metamodeling techniques in support of engineering design optimization. *Journal of Mechanical Design*, *129*(*4*), 370–380.

Wu, J. and Azarm, S. (2001) Metrics for quality assessment of a multiobjective design optimization solution set. *Journal of Mechanical Design*, *123*, 18–25.