DEVELOPMENT OF A MULTI CRITERIA PROCESS DESIGN FRAMEWORK FOR SUSTAINABLE DESIGN AND SYNTHESIS OF PROCESSES

By

MAZDAK SHOKRIAN

Bachelor of Science in Chemical Engineering
Iran University of Science and Technology
Tehran, Tehran
2005

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
December, 2012

DEVELOPMENT OF A MULTI CRITERIA PROCESS DESIGN FRAMEWORK FOR SUSTAINABLE DESIGN AND SYNTHESIS OF PROCESSES

Dissertation Approved:

Dr. Karen A. High

Dissertation Adviser

Dr. Russell R. Rhinehart

Dr. AJ Johannes

Dr. Baski Balasundaram

Acknowledgements

It is a pleasure to thank the many people who made this thesis possible. It is difficult to overstate my gratitude to my PhD supervisor, Dr. Karen A. High, with her inspiration, and her great efforts to help me. I would like to thank the many people who have taught me process design and chemical engineering: all my teachers especially graduate professors who made it possible for me to reach this point. I would like to express my special thanks to Dr. R. Russell Rhinehart for teaching me a lot in process modeling, process control and process optimization and for his kind assistance with giving wise advice, helping with various applications, and so on.

This dissertation is dedicated to my lovely and devoted parents Mr. Jamaleddin Shokrian and Mrs. Monir Shokrian for constantly supporting me throughout the beginning of my studies and believing in me and most importantly teaching me an honorable way of living. Also to my brother, Mahan Shokrian for his kind and constant supports.

Lastly this thesis would not have been completed without the constant motivation, prayers and encouragement from my cherished friend and fiancée, Nina Shakerifard.

Name: MAZDAK SHOKRIAN

Date of Degree: DECEMBER, 2012

Title of Study: DEVELOPMENT OF A MULTI CRITERIA PROCESS DESIGN FRAMEWORK FOR SUSTAINABLE DESIGN AND SYNTHESIS OF PROCESSES

Major Field: CHEMICAL ENGINEERING

ABSTRACT:

A methodology is proposed to address sustainability concerns with special intention to chemical process design. Through this methodology, aspects of sustainability are evaluated with sustainability metrics which are calculated using a sustainability evaluator module. This module is part of a novel platform for selection of most sustainable alternatives of process technologies. The platform consists of a novel rigorous multi-objective optimization algorithm in tandem with a commercial process simulator for rigorous design and optimization of the processes. The platform is capable of handling a multitude of objectives and constraints. This project develops a rigorous optimization based framework in which the ultimate goal is to select most desirable process alternatives based on a multitude of criteria such as sustainability impacts. This research contributes to sustainability development assessment in process design. ***This study addresses a major challenge in process design as engineers will now be able to design for multiple criteria, not restricted but specifically sustainability concerns. This work will develop approaches for the optimization of chemical processing technologies with sustainability considerations.*** The objectives of this research are to:

- Develop a modified Particle Swarm algorithm to handle a multitude of objectives
- Develop a method for handling mixed integer optimization problems
- Propose an effective constraint handling methodology
- Formulate an algorithm that incorporates multi-objective optimization a non-dominated methodology to enable process design with several objectives especially to satisfy social, economic and environmental objectives
- program a generic computer aided tool to assess process technologies in terms of sustainability
- Propose a decision making procedure for selecting the most sustainable process from multiple optimum alternatives that will remain sustainable in the future
- Link the proposed optimization algorithm to a process simulator for black box process optimization
- Evaluate the performance of the methodology

INTELLECTUAL MERITS: This proposed *Sustainable Computer Aided Design* framework presents the best possible trade-off between theoretical novelty and engineering applicability for long term sustainable design as well as decision-making in process engineering. The successful development of an effective and efficient multi-objective optimization tool will enable engineers to design more sophisticated products, processes, or services considering multiple conflicting objectives.

TABLE OF CONTENTS

Chapter Page

LIST OF TABLES

Table Page

Table Page

LIST OF FIGURES

Figure Page

Figure Page

NOMENCLATURE

$A \in R^m$	Feasible region
α_k	Weighting factor for objective k
cv_j	Violation of the j constraint
CR	Consolidation Ratio
$C(A, B)$	Weakly dominancy criteria for set A and B
d^i_{min}	Minimum normalized distance between particle I and leaders
d^i_j	Normalized distance between particle i and leader j
D_{ij}	Distance between *archive* member i and
d_i	Crowding distance
ϵ_i	Small positive number
f_i	Objective function
$f_p(x)$	Penalized objective function
f_{del}	A fitness value for deleting each *archive* member
f_{sel}	Success rate to keep successful members in *archive*
$f^U_{compromised,\ k}$	Compromised upper limit on objective function k
$f^L_{compromised,\ k}$	Compromised lower limit of objective function k
$f_{i,k}$	Value of objective function i for solution k.
G	Global Best in PSO algorithm
H	Number of members which share one hypercube
L^j_t	Leader j^{th} in the swarm

n	Number of leaders
$Prod_{DME}$	Production rate of Dimethylether c_1 andc_2 two positive constants
P^i_t	Personal best for particle i
P	Personal Best in PSO algorithm
$r_1$and r_2	two random values in between [0,1]
$rand$	A random number
$r_{i,j}$	Penalty parameter of the jth constraint for the ith objective function
r_i	Penalty parameter for the ith objective function
$\boldsymbol{R_i}$	Aggregated membership
$sigmoid$	Sigmoid function
$v^i_{j,,t}$	Velocity of decision variable jth of particle ith at iteration t
$v^i = (v^i_1, v^i_2, \ldots, v^i_m,)$	Vector of velocity
w_i	Weighting factors
w	inertia constant
$x^i_{j,,t}$	Decision variable jth of particle ith at iteration t
$x = [x_1, x_2, \ldots, x_n]^T$	Vector of decision variables in R^n
$x^i = (x^i_1, x^i_2, \ldots, x^i_m,)$	Decision vector in the decision variable space
$v_{ij}^{\ 1}$ and $v_{ij}^{\ 0}$	Probability of change to one and zero respectively for jth bit of the ith particle
X_i , Y_i	Discrete variables
$Z \in R^n$	Objective function space
Ω	Overall measure of feasibility
OF^i_k	Value of objective function number k of the i^{th} *archive* member
$\forall$	Mathematical symbol for "each"
$\in$	Mathematical symbol for "to belong to"
Δ	Non-uniformity measure
$\mu_{i,k}$	Membership function for solution i in terms of objective function k

ACRONYMS

ACO	Ant Colony
ANOFE	Average Number of Function Evaluation
AHI	Atmospheric Hazard Index
CWRT	Center for Waste Reduction Technologies
CRF	Capital Recovery Factor
DM	Decision Maker
DV	Decision Variables
DME	Dimethylether
DE	Differential Evolution
EHI	Environmental Hazards Index
GP	Goal Programming
GA	Genetic Algorithm
HON	Hazardous Air Pollutants
ILP	Interval Linear Programming
LCA	Life Cycle Assessment
MINLP	Mixed Integer Nonlinear Programming
MILP	Mixed Integer Linear Programming
MOO	Multi Objective Optimization
MOMLPSO	Multi Objective Multi Leader Particle Swarm Optimization

NLP	Nonlinear Programming
NSGA	Non-dominated Sorting Genetic Algorithm
NOFE	Number of Function Evaluation
NRC	Percentage of run converged to the global optimum
PSO	Particle Swarm Optimization
PSI	Parameter Space Investigation
ROI	Return on Investment
SOO	Single Objective Optimization
SIF	Social Influence Factor

CHAPTER 1

INTRODUCTION

1.1 Sustainability

A 1987 World Commission on Environment and Development report defines sustainability as "development that meets the needs of the present without compromising the ability of future generations to meet their own needs" (Brundtland, 1987).The issues driving sustainability (Figure 1.1) can be classified into: visible impacts, stakeholder demands, stricter environmental regulations, financial risks, safety and supply chain pressure (Bakshi, 2000; Beloff , 2009). Our society desires environmentally friendly processes and products that reduce resources use while considering social impacts.

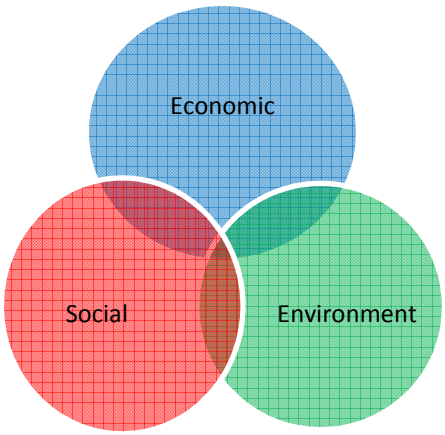


Figure 1.1-sustainability aspects (Adams 2006)

The International Council of Chemical Associations (ICCA), representing the global chemical industry, has laid a platform for moving towards becoming a sustainable sector. However, some issues have to be resolved including "continuing to evaluate alternative products and manufacturing processes, and substituting more sustainable products where appropriate" (ICCA, 2002). ***This work will develop an approach for sustainable process design and retrofit.***

1.1.1 Sustainability Approaches

As Thomas Seager (2008) and Seager et al (2007) state in triple bottom line approach for sustainability (Darton, 2003) social and environmental impacts along with profitability are the focus of multiple criteria design. The approach to solve these problems is with multiple objective optimization or multi-criteria decision making. Engineers typically align with the triple bottom line approach where quantification of the problem is possible. Problems of this sort are solved by a multidisciplinary team of experts; yet the focus may still be considered to be "efficiency", in that if money is saved, consumerism is again activated.

1.1.2 Goals and Objectives of this Dissertation

The goal of this research is to use multi-objective optimization in a systematic framework for developing new or modifying existing processes to be inherently sustainable. This methodology incorporates sustainability using economic, environmental and social metrics. Metrics capture sustainability ideas and transform them into quantitative measures that are useful for making decisions (Beloff et al., 2005). The newly developed approach for evaluating sustainability metrics by Shadiya (2010) are used to quantitatively assess the process social and environmental impacts of any process as well as profitability.

An efficient multi-objective optimization method is developed for generating and exploring all feasible configuration alternatives and operational conditions of the process in a mixed integer

solution space. The alternatives are evaluated with respect to all sustainability concerns. This research will help decision-makers determine the optimum process alternatives that maximize economic and social benefits and minimize environmental impacts. A novel Particle Swarm Optimization (PSO) algorithm capable of handling a multitude of objectives is developed. The following are the objectives of this research:

- Develop a modified Particle Swarm algorithm to handle a multitude of objectives
- Devise a mathematical algorithm to link a sustainability evaluator module and the robust multi-objective PSO algorithm for evaluating process alternatives in terms of sustainability metrics during the optimizing procedure
- Formulate an algorithm that incorporates multi-objective optimization using a non-dominated methodology to enable process design with several objectives especially to satisfy social, economic and environmental objectives
- Propose a decision making procedure to select the most preferred sustainable process alternative
- Evaluate the performance and reliability of the methodology by testing it on a Dimethyl ether process improvement

The following are the **outcomes** of this research:

- An optimization tool that will enable engineers to design and modify sophisticated products, processes, or services considering multiple conflicting objectives
- A convenient and reliable framework used for making profitable business decisions

This research is **unique** in that it will develop:

- A novel time-efficient PSO based algorithm to obtain the Pareto front of a multitude objective

- A framework to handle a high number of sustainability concerns with an evaluator programmed in MATLAB in tandem with a sequential process simulator

This project addresses the following two green chemistry and engineering concepts: "*Prevention*-It is better to prevent waste than to treat or clean up waste after it has been created and *Design for Energy Efficiency*-energy requirements of chemical processes should be recognized for their environmental and economic impacts and should be minimized (Anastas and Warner, 1998)" This project also addresses six of the Sandestin (U.S Environmental Protection Agency, 2006) green engineering principles which are: "conserve and improve natural ecosystems while protecting human health and well-being, use life-cycle thinking in all engineering activities, ensure that all material and energy inputs and outputs are as inherently safe and benign as possible, minimize depletion of natural resources, strive to prevent waste, develop and apply engineering solutions, while being cognizant of local geography, aspirations, and cultures and create engineering solutions beyond current or dominant technologies; improve, innovate, and invent (technologies) to achieve sustainability."

This work provides an effective methodology that uses multi-objective programming and decision making. The use of this framework during design enables engineers to create sophisticated products, processes, or services considering multiple conflicting objectives. The results have special applicability in sectors that process natural products and have substantial variation in material properties and rely on subjective models of process behavior such as the mineral, food, pulp, chemical and petroleum processing industries. The results may also have applicability to emerging industries in which processes are characterized by incomplete knowledge. The application of this framework extends to handling multi-criteria process design and it is not restricted to design for sustainability. ***This project will address a major challenge in the processing sector as engineers will now be able to handle multi-criteria decision making such as in the design for sustainability.***

1.2 Research Strategy

This section outlines the main research challenges and the research strategyof this work and the procedures that was pursued to develop a methodology to find the most sustainable alternative or modification for a process.

The main question and challenge addressed in this work is which optimization methodologies and algorithms are the most appropriate for creating the non-dominated set of alternatives? In order to generate all the possible configurations of the given process systematically, the superstructure representation approach (presented in Yeomans and Grossmann, 1999; Biegler et al., 1998) will be used. Because the efficiency of optimization techniques depends greatly on the nature of the problem, both conceptual and algorithmic aspects will be analyzed simultaneously. In a superstructure optimization, non-linear objective functions and constraints include both integer and continuous variables. Continuous variables account for operational conditions and integer variables account for process configuration.

The second important challenge which is addressed in this work is how to evaluate process alternatives reliably and in such a way that simplification of process design does not compromise on optimality and reliability of the solutions.

Another important addressed here is how will an understandable solution set be presented to the decision-maker? Fuzzy logic based methods are used to reveal essential characteristics of the optimal solutions which have been previously generated. The decision maker (DM) is involved after non-dominated alternatives have been introduced as the final solution set.

The methodology involves implementing a stochastic based optimization algorithm which is capable of handling combinatorial search space. This algorithm interacts dynamically with ASPEN Hysys, a sequential modular process simulator. To evaluate the process alternatives interactively, sustainability metrics are evaluated by a sustainability evaluation module linked to

the robust multi-objective optimization algorithm. And finally a decision making step is applied to reduce the cognition load on the decision maker. ***Using the proposed multi-objective optimization based methodology will be more convenient for engineers to design or modify processes that are economically and environmentally friendly and that benefit society.***

1.3 Test Methodology

The developed optimization algorithm and the proposed framework are tested on several benchmark optimization functions and a chemical process design problem. Performance of the algorithms is compared using performance metrics recommended in literature (Bartz-Beielstein et al. 2003). An exploration is done to compare the performance of the new algorithm with currently existing algorithms. Next the methodology will be implemented on the dimethyl ether process. This process is regulated under the Hazardous Organic National Emission Standards for Hazardous Air Pollutants (HON). The goal is to implement the proposed methodology and demonstrate that the overall sustainability impact of these processes has improved. The criterion for success of this objective is that the methodology is consistent and applicable to any process.

1.4 Summary

It is no longer appropriate to evaluate processes for economic feasibility alone; social benefits and environmental impacts must be considered. This research contributes to sustainability development in industry as benefits such as reduced pollution minimize resource usage and more economic products will be thoroughly explored. The application of this research extends well beyond the sustainability considerations to very vast area of optimum design with multi-criteria. The application extends beyond the chemical and petroleum industry and into other industries such as mining, energy, pulp, paper and pharmaceutical with regard to an enormous variety of decision-making scenarios as well as any kind of technology. ***A decision making framework will***

be developed that will allow for the selection of the "best" alternative among a choice of multiple technologies.

Application of the methodology could have significant reduction of impact on the environment, especially those emerging long-term environmental effects. ***This research will promote the transition from traditional emission-oriented to impact-oriented industrial pollution prevention.*** Furthermore, this tool enables engineers to conveniently design optimum processes considering more than one objective not restricted to sustainability concerns.

In conclusion, the optimization of chemical processes for sustainability with environmental, economic and social considerations results in robust engineering designs.

1.5 Dissertation Organization

Each chapter of this dissertation is an independent research study that develops a part of the overall framework Therefore, the relevant literature review is provided in each chapter to make it easier for the reader to follow the challenges and contributions of the particular piece of the work regarding the current state of the art. The organization of chapters of this dissertation is presented in the following table with a short description of the information presented in each.

Table 1.1-Organization of the dissertation chapters

Chapter number - name	Information presented
1 -Introduction	This chapter elaborates on the motivation for this work and rationalizes the goals and the steps in the development of this framework
2 -Systematic Sustainable Processes Development	This chapter reviews past and current trends in process system engineering and the integration of sustainability. The proposed features of this frameworkand how it advances the work in this field are explained.

3-Sustainability and Process Evaluation in Terms of Sustainability	The basis and state of the art for sustainability evaluation using quantitative metrics are presented in this chapter.
4- Optimization and Its Application in Process Engineering	Process optimization methodologies and recent advancements are reviewed and discussed briefly in this chapter. Also, the need for development of a new optimization algorithm is presented.
5- Novel Multi-Leader Multi-Objective PSO Algorithm (MLMOPSO)	The steps of development and structure of a new optimization algorithm called MLMOPSO is presented in this chapter a long with results and discussion of its application on well-known optimization problems.
6-Multi-Criteria Process Optimization Framework	This chapter is devoted to explanation of the developed framework, its modules and application of it on a chemical process synthesis problem.
7-Conclusion and Future Work	The implications and impact of the application of the developed framework are concluded in this chapter and directions for future work and studies are provided.
Appendices A and B	Sustainability tables, Computer programs

CHAPTER 2

SYSTEMATIC SUSTAINABLE PROCESSES DEVELOPMENT

Process synthesis has been in the center of research focus for several years due to its importance and complexity. Synthesis is complex because the problem to solve is under defined and only a fraction of the necessary information for design is provided by chemists and physicists. Assumptions must be made to complete the synthesis task. Thus, normally there are numerous ways for a specific synthesis task (10^4 to 10^9 ways) (Douglas, 1988). Traditionally, the only criterion for selecting one of the possible methods or process alternatives was to select the most profitable process. However, a process also needs to be safe, flexible and satisfy other important criteria such as environmental and health impact.

Regardless of the type of processes, the synthesis procedure and steps are common to all types of process design. Motard and Westberg (1978) proposed an applicable generalization that to this day is still relevant. Based on their generalization, process synthesis bottlenecks and complexities can be categorized into three main categories:

- Representation of process alternatives in such a way that includes all process alternatives and allows elimination of unreasonable alternatives and selection of the most desirable alternative according to a set of criteria

- Evaluation of each process alternative based on a set of criteria and then comparison of the alternatives
- A strategy to effectively locate the optimum process alternative without enumerating all possible alternatives

Generally process synthesis methodologies can be categorized in three main and vastly different categories as: heuristic methods, optimization based method, and hybrid methods(Li X. and Kraslawski A., 2004).

2.1.1 Heuristic Methods

Heuristic methods are defined as procedures for process synthesis which utilize implicit long term engineering knowledge combined with explicit engineering rules. Pahl et al., (1996) state that heuristic methods are necessary for organizing the sequence of operations. Douglas (1988) has proposed a heuristic method for synthesis which is a modified version of pure heuristic approaches, called hierarchical process design that is still relevant today. In this method, long term engineering experience is applied at different level of process synthesis to generate possible alternatives. At each level more information is available for the next level and promising process alternatives are generated based on economic criteria. Short cut engineering calculations are used for simplification and estimation in this methodology. At the last step of this procedure a more rigorous overall calculation is done to check the consistency of the previous calculation and modify some assumptions. The hierarchical steps (different levels) can be listed as follow (Douglas, 1988):

- Process type, batch versus continuous
- Input-output structure of the process and overall material balance
- Recycle structure and reactor design
- Separation system design and rigorous material balance

- Heat exchanger network design

In the hierarchical process synthesis the process is decomposed to several levels and at each level alternatives are generated and evaluated in terms of economic criteria. However, this method does not look at a process as an integrated unit and there possibility to miss the interaction of different levels of the process. This in some cases would results in not finding the optimum process alternative. Other researchers also presented similar heuristic methodology for process synthesis (Smith and Linnhoff, Smith, 1995, Schembecker and Simmrock, 1997).

The mentioned methods consider economic criteria for process design.However, due to the importance of other aspects such as environmental and social impacts, the heuristic methods have been extended by other researchers. Two extended heuristic methods have been proposed considering pollution prevention by Pennington (1997) and Butner, (1999) that address these additional concerns. However, there are some important limitations associated withall heuristic methods (Carvalho, 2009):

- Heuristic methods are experienced based methods which structurally use engineering knowledge and shortcut and simplified calculations. In some cases this would result in missing optimum alternatives due to lack of in-depth design.
- The heuristic methods are imperfect because they are based on prior knowledge and may miss some possible interactions between different parts and levels of a process.
- These methods sometimes compromise on the optimality of the final solution due to simplifications and assumptions.

2.1.2 Optimization Based Methods

In the optimization based approaches all of the alternatives of a process are presented in a process superstructure. The process superstructure consists of all possible sequence and types of operations, connections, and pathways which could fulfill the ultimate process task. The synthesis of the process is formulated as an optimization problem to systematically find the optimum alternative among all alternatives presented in the superstructure. Formulation of a superstructure needs the use of mixed integer programming which most often is a highly nonlinear program due to non-linearity of process design equations. Solution of a mixed integer non-linear/linear optimization problem (MINLP/MILP) requires efficient numerical solvers and advanced mathematical programming techniques (Li X. and Kraslawski A., 2004).

Meta heuristic and stochastic optimization methods have been developed and used for process synthesis in the past two decades as well as traditional mathematical programming (MINLP). Regardless of the optimization approach the superstructure of the process is formulated as an optimization problem. Subsequently the optimum is found using an optimization algorithm. Many studies have applied the optimization base approach for process synthesis. Grossmann and Daichendt(1996) reviewed the optimization based methods in process synthesis. The main advantage of this approach is that a variety of process synthesis problems can be handled systematically and more rigorously in terms of process design and interactions based on a criterion such as economic performance. The main disadvantage of the optimization methods are their computational burden in solving a very large set of nonlinear equations. Due to the fact that the formulation of the whole process is too complex, methodologies have been developed to focus only on synthesis or improvement of a specific part of the process, but not both. Gundersen and Grossmann (1990) have proposed a systematic methodology for the optimization of heat exchanger networks. Ponce-Ortega et al.(2008) developed a new formulation for heat exchanger network modification. They considered the interaction between process conditions to select the

optimum alternatives of heat integration options. However all existing optimization based methods share disadvantages as follows (Carvalhoe, 2009):

- Involve complex mathematical formulation which is not conveniently tractable for regular process engineers and require experts to formulate and solve them.
- Restricted to the type of processes that each method has been developed for. Usually the existing methods are not extendable to other processes and they are not generic.
- A huge computational burden and long development time to for the mathematical representation.
- In most cases, the inability to consider multiple objective process synthesis efficiently.
- Utilizing simplified formulation of the process to reduce computational effort and make the problem tractable and thus not being completely rigorous.

2.1.3 Hybrid Methods

As mention in the previous section, optimization based methods are based on a superstructure representation of processes. Thus still there is a need for generating process alternatives. Hybrid methods combine a variety of approaches and knowledge about process design to generate alternatives and synthesize a process which creates applicable and power fool tools for process synthesis. Fischer et al.(1987) proposed a systematic method for screening process alternatives. This approach is based on a sequence of heuristic rules and optimizing the process at each level of synthesis based on operating cost of the process. Dantus and High (1996) also proposed a frame work for flowsheet configuration using a hybrid method. In their method the superstructure of processes is built through a sensitivity analysis and a hierarchical method and then the superstructure is optimized using a stochastic optimization approach.

2.2 Current Trend in Process Design and Synthesis

During past decades process synthesis has been the focus of researchers and has been developed to a high level of maturity. However, better methodologies need to be developed to further incorporate the social and environmental concerns into process design. Economic, environmental, safety and social metrics of sustainability are required to be addressed in development of processes to avoid compromising the need of future generations while meeting the need of present generation and market demands. Sustainability concerns have changed the trend in process design and synthesis intensely. The capability of design for sustainability differs tremendously with the level of process design. The earlier the level of process synthesis, the easier is to design for sustainability and the more the degree of freedoms are available. Therefore, the current challenges in process synthesis falls in two main categories: one is to incorporate the concerns of sustainability quantitatively using a generic applicable methodology and another is to develop applicable tools which enables process engineers to conveniently incorporate the sustainability concerns in early stages of process synthesis and process improvement. This requires of handling multi-objective optimization problems and evaluation of process alternatives rigorously to select the most sustainable design (Li X. and Kraslawski A., 2004).

Although the major concerns of sustainability are well elaborated, still there is a need to incorporate them in to process synthesis. Thus, complicated decision-making scenarios may arise from a large number of design objectives with constraints that have to be simultaneously satisfied. When engineers are confronted with sustainability, it is not clear which methodology or tool they should use. None of the current tools for optimization present a comprehensive methodology to consider all concerns of sustainability as separate objectives. Existing methodologies are based on equation oriented process simulation, an approach that is not convenient and is not rigorous in terms of simulating the behavior of real processes, and are not

suitable for multi-objective optimization. Thus, there is a need to develop a systematized design and modification methodology that considers conflicting sustainability criteria using a sophisticated optimization and decision making approach (Grossmann and Guillen-Gosalbez, 2009).

When considering sustainability concerns as the criteria for design or modification of processes using a multi-objective method, three issues are encountered: first, construction of a framework for exploration and evaluation of process alternatives rigorously using a multi-objective optimization algorithm (Li X. and Kraslawski A., 2004); second evaluation of process alternatives rigorously; and finally selection of the most preferred solution among all non-dominated solutions.

2.3 Related Work Advantages and Disadvantages

During the past decade most researchers have been trying to address the mentioned issues and develop frameworks which address the two main challenges encountered in incorporating sustainability concerns:

- Sustainability evaluation
- Multi-objective optimization

2.3.1 Sustainability Evaluation

Incorporating the sustainability criteria into process design requires the evaluation of the sustainability concerns quantitatively as well as the use of a generic and unified approach. Minimizing waste generation, environmental penalties (global emission) (Cabezas and Douglas, 1999), mass of pollutant of concern, and total mass of waste (Ahmad and Barton, 1995) have been often used as representative of environmental concerns of sustainability in 1990's (Li, C., et

al.2009). However, due to the involvement of energy consumption and the effect of recycle flow along with materials in a process, the use of traditional methods are not reasonable. In addition social impacts and safety of processes are very important to be considered in process synthesis.

To overcome the challenge of incorporation of sustainability into process design and decision making, several metrics have been proposed to guide sustainable decision making (Tugnoli et al. 2008; Azapagic et al. 2004; Beloff et al. 2005; Beloff et al. 2002; Martins et al. 2007; and Sikdar, 2009). Examples of metrics proposed by these researchers include: material intensity, energy intensity, water consumption, toxic emissions, pollutants emission and potential chemical risk. Recently, some methodologies also have been presented for evaluating the social and environmental impact of processes. Examples of these methods are: life cycle assessment (LCA) (Lou et al., 2004), the environmental fate and risk assessment tool (Achour et al., 2005), the atmospheric hazard index (AHI) (Gunasekera and Edwards, 2003), thermodynamic analysis method (emergy and exergy) (Hau et al., 2007; Yi and Bakshi, 2007; and Bakshi, 2002). The reader is referred to Chapter 3 of this dissertation for a detailed review of sustainability evaluation methods.

Although researchers have put forth efforts towards quantifying sustainability, there are several limitations in existing methods. One of the important drawbacks is that existing metric systems stress noticeable or foreseeable changes resulting from human actions without considering long term changes that are typically associated with sustainability. Reliable approaches that address this issue are needed. So the assessment of sustainability metrics quantitatively to be integrated in process synthesis is another complexity which needs to be addressed.

2.3.2 Multi-Objective Optimization and Incorporating Sustainability into Design

Process synthesis and design is involved with nonconvex combinatorial optimization problems, thus solving nonconvex combinatorial decision making problems within a reasonable amount of time is a necessity for flowsheet optimization and process synthesis. In addition, multi-objective optimization algorithms are required which are capable of handling a multitude of objectives in combinatorial search spaces and obtain diverse, good quality solution sets for a complicated process.

There have been many great achievements using analytical optimization algorithms (e.g. Grossmann and Biegler, 2004; Sawaya, 2006) such as interval linear programming (ILP), chance-constrained programming, and mixed integer-linear and nonlinear programming (MILP) and (MINLP) techniques have been applied to find solutions which meet the sustainability criteria(Cai YP, 2009) and also attempts to develop deterministic based global optimization (Floudas et al. 2005; Karuppiahand Grossmann, 2008; Lundell, Westerlund and Westerlund,2009; Tawarmalani and Sahinidis, 2005), graph-theory based algorithms (Brendel, Friedler, and Fan, 2000), multi-objective integration of safety and process design (Kim, Yeo, and Moon, 2004), process control and design (Miranda et al. 2008) enhanced capabilities of process optimization and synthesis. However, these methods are not well suited for incorporating sustainability criteria into process design. The main reason in that analytical approaches are very inefficient in handling a multitude of objectives and it is not an easy task to combine sustainability criteria to only one objective and this would not result in obtaining true solutions of the problem.

Analytical based global optimization methods are very dependent on the complexity of the process and are not suitable for dealing with the multi-objective problems which appear in sustainability. Most existing tools are based on using equation oriented process simulators which are not comparable with the sequential modular simulators such as Aspen Plus in terms of

convenience and reliability. Although Aspen Plus includes an equation oriented module for process optimization, it is limited in scope as it only handles single objective optimization with well-behaved functionality.

Alternatively, the use of heuristic and hybrid approaches for the sustainable development of processes has been on the rise. A multi-objective optimization framework for quantifying and incorporating the environmental and economic metrics in process synthesis has been proposed by Fu et al. (2000).In this method some of the environmental concerns of sustainability are posed as conflicting objectives of a multi-objective problem. Goyal and Diwekar (2001) proposed a hybrid method consisting simulated annealing and nonlinear programming to incorporate process environmental impacts into process synthesis. In this algorithm the general waste reduction algorithm (WAR), a methodology for determining the potential environmental impacts of chemical processes, is used for evaluation of environmental impacts. The capability of handling uncertainties is the main advantage of this method. However, the high computational cost and incapability of handling multiple objectives limit the application of it. A conflict based approach for handling the objectives has been proposed by Li et al. (2003) for multi-objective process optimization which extracts the conflicts of the objectives and solves them. This approach has been illustrated by its application on a process design case study by considering both economic and environmental impacts of the process (Li et al., 2003). Other approaches such as the weighted sum of the objective functions, goal programming (GP), and parameter space investigation (PSI) methods have been combined together by Young et al. (1999) to simultaneously sample the search space within a process simulator and handle multiple objectives of the problem. Kahraman et al. (2009) used fuzzy logic for multi-objective optimization of renewable energy alternatives. Connolly et al. (2010) reviewed the main existing computational algorithms for incorporation of renewable energy options in various energy systems. The Non-dominated Sorting Genetic Algorithm (NSGA) algorithm has been applied during the past decade

for handling a variety of multi-objective process design such as industrial nylon-6 semi-batch reactors by Mitra et al. (1998) and a side-fired steam reformer by Rajesh et al. (2000). Soroudi et al. (2011) proposed a multi-objective method for the long term planning of distribution network expansion along with distributed energy options. This method uses an immune genetic algorithm to optimize cost and emissions having the schemes of sizing, placement and dynamics of investments on distributed generation units and network reinforcement over the planning period as decision parameters.

Methodologies developed for the sustainable design of processes use three approaches for incorporating sustainability in process design. The most common approach is to consider the environmental impacts of the process as main objectives for the optimization of processes while treating the economy, product specification and other impacts of the process as constraints or inferior objectives. Dantus and High (1996) proposed a methodology using simulated annealing optimization method to reduce the energy consumption of the process and minimize waste generation. Soorathep and Masahiko, (2004) have developed a procedure that uses a hierarchical based approach to incorporate the environmental metrics as constraints in the optimization process while considering economics as the main objective. More recently, Cave and Edwards (1997)proposed the environmental hazards index (EHI) for the process selection which however, are only suitable to be applied for early stages of process design and cannot ensure global optimal results (Chen et al., 2002).

As opposed to treating some of the sustainability criteria as inferior objectives or constraints for design, methodologies have been developed based on more rigorous optimization techniques which treat all of the objectives simultaneously with the same level of priority. These methods are more suitable for handling conflicting objectives and do not require bias toward any of design criteria. For example, a multi-objective method has been proposed by Ciric and Gu (1994) in which the waste generation is minimized while the profit is maximized. Sengupta et al. (2008)

developed a chemical complex analysis system using multi-criteria optimization to determine the optimal configuration of plants in a chemical complex based on economy, energy, environment and sustainable costs. Othmana et al. (2009) addressed the sustainability concerns using a multi-criteria analytical process hierarchy. Piluso and Huang (2009) devised a framework for industrial sustainability using forecasting and profitable pollution prevention. Other researchers have used multi-objective optimization and multiple criteria decision making to ensure sustainability (Guillen-Gosalbe and Grossmann, 2009; Hossain, 2008; McPhee and Yeh, 2004; and Sun and Lou, 2008). Li et al. (2009) have developed an environmentally conscious method using an impact assessment methodology.

This latter approach is the most powerful and promising approach for obtaining global optimum solutions to a process design problem while considering conflicting sustainability criteria. Nevertheless, the application of this method requires highly effective optimization algorithms capable of handling a multitude of objectives. Existing optimization algorithms fail to handle design problems using this approach reliably in terms of process design with a reasonable computational expense.

Researchers have put forth efforts to solve the problem of computational complexity and handling multiple design objectives by utilizing another approach which simplifies the process model or combines the objectives of design into one single objective. Ramzan and Witt have developed an efficient four stage procedure for simplifying multi-objective optimization problems to handle conflicting objectives more effectively (Ramzan and Witt, 2006; and Ramzan et al., 2008). A hierarchical optimization methodology also has been proposed by Lou and Singh (2006) which provides a systematic procedure for solving multi-objective problems. Non-dominated sorting NSGA-II algorithm has been used for multi-objective optimization due to its simplicity for multi-objective optimization applications. Inamdar et al. (2004) applied this algorithm for the multi-objective optimization of a distillation unit which shows the applicability of this algorithm

in handling complex model. However, other concerns of sustainability except economy were not considered in their work.

The main drawbacks of the studies described above could be summarized as:

- Inability to ensure the global optimum or true non-dominated solution set
- Inability to evaluate and design processes rigorously and reliably
- Difficulty in handling multi-objective problem
- Absence of a suitable environmental objective that can reflect the whole environmental impact of a process including energy and material usage

Therefore, there is a need to improve on previous work and develop an effective user friendly multi-objective optimization framework that enables multi-criteria process design and addresses the three major sustainability concerns.

2.4 Problem Specification and Solution Approach

2.4.1 Current challenges in systematic process development

As it has been elucidated, multi-criteria process synthesis is a need in the processing industry due to the importance of other criteria such as environmental and social impacts of processes beyond the typical consideration of the economy of the process. There are a multitude of sustainability metrics which cover all economic, environmental and social aspects of a process. However, most of the sustainability metrics are not combinable. Normalizing sustainability concerns and combining them as a single criterion is not an easy task and is impossible in most cases. Thus, a process synthesis framework based on a multi-objective optimization method is a need. Also evaluation of sustainability at each stage of design especially during the early stages is very important to. Failure to accomplish this will result in an unsustainable process.

On the other hand, as mentioned in Chapter 2, there exist two other issues which have not been well explored in this field. First the processes need to be designed and evaluated as

integrated units rigorously to generate reliable results. Also analyzing a high dimensional non-dominated solution set is very cumbersome and which should be taken into account while dealing with intricate multi-objective problems. These two issues also need to be addressed by developing new frameworks.

Considering new concerns imposed by society such as sustainability, concerns which cannot be transferred to economic terms easily, dictate the need for reliable design and a reduction of high cognition load on decision makers. New problems are proposed to contribute to the recent trend and challenges in process synthesis for systematically exploring and evaluating process alternatives. These problems listed below should be addressed as well as the previously mentioned problems of incorporating a multitude of sustainability concerns into design of complex processes.

- Assessing process alternatives in terms of sustainability concerns quantitatively
- Multi-criteria decision making for selection of the best alternative based on several criteria
- Reliable process evaluation to avoid compromising the process characteristics for simplification of process design problem
- Development of systematic framework which enables engineers to synthesize processes conveniently and reliably
- Presentation of the non-dominated solutions to the decision maker in such a way that reduces the cognition load on the decision maker which is due to presence of a multitude of objectives.

Recently, Li C. et al. (2009) developed a new methodology for integration the sustainability criteria into the design and synthesis of processes which overcome some of the mentioned challenges. In their study, an environmental impact assessment method is presented based on an internal data base. Environmental impacts of chemicals and energy are characterized and assessed

by their proposed methods to be used as guidance for optimization of environmental benign processes. They use a process model to design and evaluate each process alternative. The process model and multi-objective optimization algorithm which is non-dominated sorting genetic algorithm NSGA, interactively generate a set of non-dominated solutions for the problem. They also have developed a decision making step in which, the preferences of the decision maker are adopted for tradeoff between solutions (Li C. et al.2009). Interestingly, this methodology applies different modules to address the mentioned issues such as quantitative assessment of sustainability metrics and reducing the cognition load on the decision maker. However, this methodology still needs more enhancements to solve some important problems such as problem specificity of the process model, inefficiency of the optimization problem in handling multi-scale and multi-dimension problems and the failure to incorporate social aspects of sustainability. Also the process model the work of Li et al. (2009) has been simplified using thermodynamic insights which may cause reducing the integrity of the problem and missing some promising solutions.

2.4.2 Solution Approach

A generic framework is presented in this PhD dissertation for systematic process design and improvement which enables the synthesis of processes based on a multitude of design criteria and allows for reliable and in-depth process sustainability evaluation. This framework enables process engineers to handle process design synthesis and decision making with multiple criteria conveniently. An important application of this framework which solves mentioned challenges in the state of the art is to incorporate sustainability concerns to process design using a generic and unified methodology.

The proposed solution framework is based on hybrid process synthesis approaches explained in earlier sections. This framework consist of two main levels; first, use of the hierarchical method proposed by (Douglas, 1988) to identify possible process alternatives for a specific process design or improvement of an existing process which are simulated as a superstructure in a

rigorous process simulator; second, formulation of the superstructure as an optimization problem and optimizing the superstructure to select the most promising process in terms of a set of pre-specified design criteria such as sustainability concerns.

The proposed framework provides the following features to solve the current problems in the field of process synthesis.

- Represents all possible process alternative identified by the hierarchical synthesis approach in a rigorous process simulator to enable in-depth process evaluation.
- Incorporates sustainability concerns into design to evaluate and compare process alternatives in terms of sustainability metrics including economic, environmental, safety and health metrics usinga sustainability evaluator quantitatively.
- Explores all presented alternatives in the superstructure using a timely efficient optimization algorithm capable of handling MINLP problems imposed by the superstructure representation and multiple objectives imposed by sustainability concerns.
- Presents a set of non-dominated solutions to the decision maker for further analysis.
- Applies a fuzzy logic based method to reduce the cognition load on the decision maker

Also the framework being presented in this PhD work has the following advantages over existing heuristic synthesis methods in that it:

- Effectively locates the optimum process alternative without numerating all possible alternatives
- Treats the process as an integrated unit as opposed to decomposed process levels to ensure accounting for all interaction between different process levels.
- Utilizes in-depth process simulation to evaluate process alternatives as opposed to using shortcut methods. Therefore, ensures not missing the global optimum solution(s).
- Does not compromise on precision of the optimal solutions due to the complexity of solving process design equations.

- Enables incorporating several design criteria to process design simultaneously.

The advantages of the proposed framework comparing to existing optimization based methods can be listed as following in that it:

- Is generic and extendable to any type of processes.
- Does not require the solution of a huge set of equations.
- Handles optimization problems with a multitude of objectives.
- Does not require any kind of simplification on design equations, and thus, does not compromise on the optimality of the solutions.

2.4.3 Overall Structure of the Framework

A systematic framework proposed for optimal process design that considers sustainability criteria and is shown in Figure 2.1. In this framework, a novel stochastic optimization algorithm is use which is explained in subsequent chapters. This algorithm efficiently handles multi-objective MINLP problems essential in optimizing a process superstructure for sustainability.

The proposed optimization algorithm interacts with a sequential modular process simulator to find the optimum configuration and operational conditions of a given process. This removes the need of formulating the whole process using mathematical models and solving all process design equation which is very burdensome. As a result, process models are not simplified; this in turn increases the reliability of the framework. On the other hand using a rigorous process simulator would provide more convenience for the designer. However, the use of a process simulator requires using an optimization method capable of handling black box optimization. This is elaborated on Chapters 4 and 5 in more details.

During the optimization procedure, the optimizer needs to evaluate the objective functions at each iteration. To do so it communicates decision variables (DVs) with the process simulator which simulates the process based on new DVs. The results of the simulator which are very

accurate are used to evaluate the objective functions and are passed to the optimizer again. In order to design for sustainability the evaluation of objective functions is performed by a sustainability evaluator module which is programmed in MATLAB. The sustainability evaluator module uses the sustainability metrics presented by Shadyia (2011). Finally, a non-dominated solution set is presented to the decision maker. The decision making module of the framework uses a fuzzy logic based method along with the decision maker preferences which are taken based on the available solutions to select one of the solutions as the best preferred non-dominated solution. The criterion for success is that our methodology shown significant sustainability improvement compared to the base case and the proposed framework is applicable to any process.

2.5 Summary of the Modules of the Framework:

In the chapters 5 and 6 of this PhD dissertation, the development of the following modules of the proposed framework is explained in details and the performance of the overall framework is tested on improvement of the dimethyl ether production process.

- A sustainability evaluator module which interacts with the optimizer to evaluate alternatives in terms of sustainability metrics.
- A timely efficient optimization algorithm capable of handling a multitude of objectives of constrained MINLP problems.
- A superstructure representing all possible optimization process alternatives which is simulated in a sequential modular process simulator allows exploring and optimizing the process alternatives.
- A decision making module which takes the preferences of the decision maker based on the available non-dominated solutions and selects the most preferred non-dominated solution.

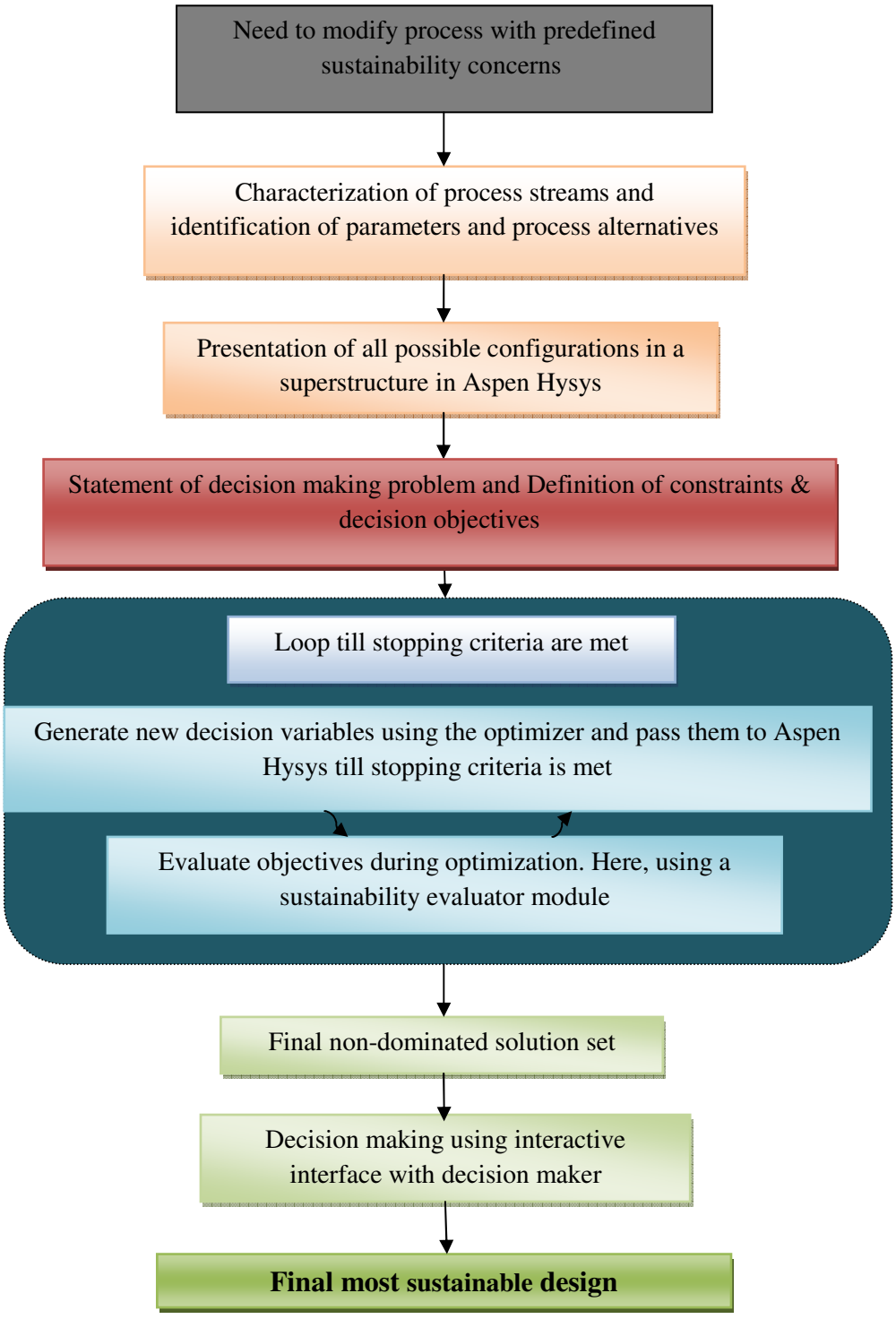


Figure 2.1, Flow diagram for overall methodology

CHAPTER 3

PROCESS EVOLUTION IN TERMS OF SUSTAINABILITY

When it comes to comparing several choices, regardless of the comparison concept, there is always a need for some type of scale or measure to be able to evaluate the choices. The main characteristic of these measures is the need to be generic and uniform. The same is true for process design decision making problems. In chemical process design and synthesis, regardless of the method that is used, the designer and the decision maker must always decide between a variety of process alternatives and technologies based on some consistent and uniform measures. The measures themselves or sometimes a form of the measure is considered as the process design criteria. As described earlier, traditional measures for comparing the process alternatives are indicators of the process economy. Each process alternative which has a better value of the economic indicators is selected over other alternatives. In this case the criterion for design is the economy of the process. However, nowadays economy cannot be the only criterion for process design as it does not reflect the other aspect of current and future society needs.

In order to truly cover all impacts to society and to the environment, processes need to meet other criteria such as safety, environmental impact, and health impact. These concerns are indicated and evaluated by different metrics of sustainability serving as criteria for synthesis and design of processes. These metrics or a form of them pertaining to each aspect are called sustainability criteria.

The concept of sustainability and sustainability metrics is elaborated in this chapter and the metrics which are used for process evaluation in this work are introduced as well. All the methods and information required for the sustainability evaluation of process alternatives are incorporated by the sustainability module of the framework and this approach is presented in this chapter.

3.1 Sustainability Concept and Sustainability Evaluation Approaches

Although the sustainability development first appeared in 1970 (need a reference here), the Brundtland Report (1987) was the commencement of the appearance of the sustainability concept in industrial design and applications. The Brundtland Report, (1987), presented sustainable development as:

> *"Sustainable development is a development that meets the needs of the present without compromising the ability of future generations to meet their own needs".*

Nevertheless, while this definition clarifies the concept of sustainability, it does not provide any applicable guideline to incorporate sustainability in practice. Many researchers and organizations have put forth efforts to reconfigure the sustainability concept to better understand the sustainability concept and develop measures and indicators for sustainability using different approaches: Frameworks, indicators and metrics (Carvalho, 2009)

Each approach has different characteristics in terms of data intensity and robustness. The methods with less data intensity require less data to use them. The framework approach has the least data intensity but also least robustness comparing to the other two. The metrics approach on the other hand has the most data intensity while it is also the most robust approach (Carvalho, 2009).

3.1.1 Framework Evaluation Approach

A sustainability evaluation framework is a generic methodology for evaluating the level of sustainability or to measure the progress toward sustainability. These methodologies use the principles of sustainability and design to rank the design actions. Development of frameworks needs wide access to international standards thus they have been developed mostly by companies and international organizations. The ultimate objective of sustainability frameworks is to generically evaluate the progress toward sustainability (Carvalho, 2009). There are several frameworks have been developed in past decades by organizations and researchers such as the Brundtland Report developed by in 1983.

3.1.2 Indicators

Indicators are used to indicate the progress toward sustainability especially over a period of time. They must have four main characteristics:

- Relevancy to the community
- Convenience to be used
- Indication of long term impact
- Interaction between three dimension of sustainability

Based on the three dimensions of sustainability shown in Figure 3.1, there are three types of sustainability indicators; economic, environmental and social indicators.

Economic indicators measure the economic progress over a period of time. Net job growth, employment diversity, number of jobs with benefits, work required to support basic needs are some examples of economic indicators (Shadiya, 2010).

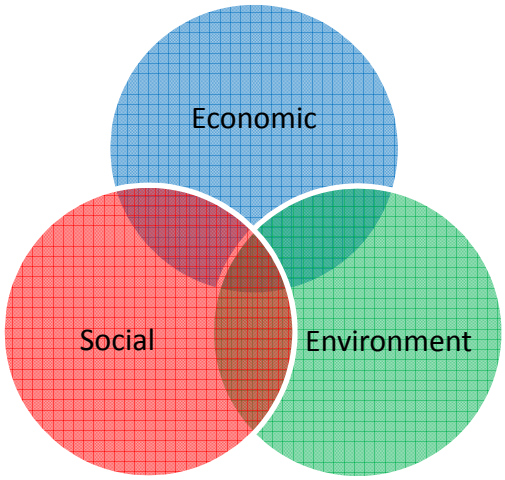


Figure 3.1-three aspects of sustainability (Adams, 2006)

Environmental indicators show the positive environmental improvement in society. Again they are measured in a period of time to evaluate the environmental progress. Ecological health measures, water quality improvement measures, air quality, criticism number, ppm of particulate material in the air, are some examples for environmental indicators (Shadiya, 2010).

Issues related to well fare of the society are addressed by the social indicators. They measure safety, health and well-being of the society over a period of time. Anderson et al., (2001) provide some example of social indicators as following:

- Number of health issues as a result of environmental pollutants
- Number of students that are inform of environmental issues in environmental education classes
- Number of community members addressing environmental issues such as global warming
- Number of families who are living below the poverty line

There are several researchers who have been working on developing sustainability indicators. Hueting and Reijnders, 2004, proposed several strategies for building sustainability indicators. Tanzil and Beloff, 2006, reviewed the sustainability indicators and metrics developed by organizations and companies. All the studied in this field are based on triple bottom line of

sustainability. The triple bottom line of sustainability shown in Figure 3.1 presents the main aspects of sustainability which need to be addressed by apposite indicators.

3.1.3 Metrics

Various aspects of sustainability are described by metrics. Metrics are built using a collection of indicators or cumulative indices (Carvalho, 2009). Metrics must have number of characteristics in common and compliance with these characteristics plays a significant role in selection of proper metrics for a certain process (as they are selected for the case study in this work). As suggested by Atlee and Kirchain, (2006) and mentioned in (Shadiya, 2010) some of these common characteristics are listed below:

- Simplicity, they should not require a long procedure or a lot of calculation to be obtained.
- Predictive, generic and consistent for different applications
- Useful as decision making tools
- Economical efficient: data collection should be easily
- Unbiased and robust to show the actual progress in terms of sustainability
- Applicable to several processes
- Applicable beyond the battery of a process addressing the supply chain issues
- Comprehensive and understandable for a variety of people

As elaborated in previous chapters, the incorporation of sustainability aspects; economic, environmental and social are essential in every step of the process design. Metrics are very proper tools for evaluation and comparison between process alternatives. As an example an economic metric like Return on investment, ROI, is calculated for each alternative and the one with higher value of ROI is selected over the others due to having a better sustainability metric. The decision maker in this manner does not have to go through complicated procedures.

Many efforts have been done in developing sustainability metrics which address different aspects of sustainability and preserve the principles of green chemistry and process and product development. The Center for Waste Reduction Technologies (CWRT) developed a set of metrics which indicate the normalized intensity of production of manufacturing. This set of metrics includes material intensity, energy intensity, water consumption, toxics dispersion and greenhouse gases (Beloff, et al., 2001). IChemE, Britain's Institution of Chemical Engineers, has recently extended the sustainability metrics to involve subsets of economic and societal indicators (Azapagic et al., 2002).The American Institute of Chemical Engineers/ Center for Waste Management (AIChE/CWRT) (2000) in 1999 developed sustainability metrics which measure the environmental impacts including material intensity, water intensity, energy intensity, toxic release, solid waste and pollutant emission. A set of socio-eco-efficiency metrics was developed by BASF to indicate the impact of process and product development in the company and includes raw materials consumption, energy consumption, land use, air and water emissions, solid waste, and potential toxicity (Saling et al., 2002). Many other systems of sustainability metrics and index have been developed by other researchers and organizations which are described in more details by Shadiya (2010) and a summary of these metrics and sustainability indices are summarized in Table 3.1.

Table 3.1 -Summary of the proposed Metric, Indicator and Index Systems presented in Shadiya (2010)

Name of the system	Application	Drawbacks
Sustainable Process Index (Krotscheck and Narodoslawsky, 1996)	Applicable to detailed process design	* Lack of incorporating reaction efficiency. * Inapplicable to early stages of design * Not addressing the social impacts
Inherent Process Safety Index (Heikkila, 1999)	Applicable for assessing the safety of a chemical process	* Limited to safety concerns

Sustainability Indicators (Afgan et al., 2000)	Assessment of sustainability at early stages	* Limited to energy indicators
ALCHE/ CWRT Sustainability Metrics (AIChE Center for Waste Reduction Technologies (CWRT), 2000)	Comparison of environmental impacts	* Limited to environmental aspect. * Metrics are accumulated into one metric.
BASF Socio-Eco-efficiency Metrics (Saling et al., 2002)	Useful in evaluate the impact of products and process during detailed design	* Limited to early stage of design due to extensive data requirement. * The social metrics impose problems in correlating process design parameters.
Green Metrics (Constable et al., 2002)	Evaluation of reaction efficiency	* Addressing only resource efficiency but not sustainability
IChemE Sustainability Metrics (IChemE Metrics, 2002)	Assessment of the sustainability of production processes	* Difficulty in correlating process parameter to the presented social metrics
Indicators of sustainable production (Krajnc and Glavič, 2003)	Assessment of the sustainability of an operating unit	* Not all of the suggested metrics are applicable for early stages of design
Global Environmental Risk Assessment (GERA) Index (Achour et al., 2005)	Applicable for addressing health and safety risks of an operating unit	* Limited to risk and safety assessment.
BRIDGES to Sustainability Metrics (Tanzil and Beloff, 2006)	Comparison of environmental impact of chemical processes	* Limited to environmental aspect. * Metrics are accumulated into one metric.
Three Dimensional Sustainability Metrics (Martins et al., 2007)	Evaluation of the sustainability of an industrial process	* Lack of direct correlation between operating conditions, chemical process risk and environmental impact in the two presented metrics
Sustainability Indices (Tugnoli et al., 2008b)	Evaluation of the sustainability of process alternatives	* Not applicable to early stage of design
AIChE Sustainability Index ("AIChE Sustainability Index: Strategic Commitment to Sustainability," 2008)	Comparison of different companies' performance	* Lack of quantitative metrics * Not applicable to early stages of design

Sustainability impacts have been presented in form of relative amounts. Metrics for environmental and economic aspects are usually a ratio of the impact over a form of total

consumption or usage. Social metrics developed by the Institute of Chemical Engineers are presented in form of percentages of payroll expenses, working hours lost as a percent of total working hours, and number of complaint per unit value added (Beloff et al., 2005). However, most of the sustainability metrics and indices do not represent social impacts and only presents the environmental and economic impacts. Due to the importance of social aspect of sustainability developing comprehensive social metrics is necessary.

3.2 Incorporation of Sustainability in the Proposed Framework

The objective of this research is to provide a generic framework which incorporates all three aspects of sustainability for use in process synthesis reliably and rigorously. In order to achieve this goal it is necessary to evaluate process alternatives in terms of all sustainability metrics which cover all the aspects of sustainability. These metrics need to comply with all previously mentioned characteristics for metrics and also they need to develop in such a way which enables assessment of the alternatives interactively and systematically during the optimization procedure. Subsequent sections of this chapter presents the sustainability metrics incorporated in the framework proposed in this PhD work and the strategy for evaluating the sustainability metrics systematically.

3.2.1 Sustainability Metrics Evaluation

Shadiya (2010) has developed a sustainability evaluator which is an Excel based tool. This tool incorporates indices and metrics from other sources to address sustainability aspects and enables evaluation of processes in terms of forty sustainability metrics, shown in Table 3.2. These metrics are calculated based on process operating conditions, process structures and material and operational costs. This tool is one of the most comprehensive tools which quantitatively evaluate

sustainability of processes by generic and uniform metrics. Some of the metrics evaluated by this tool are listed below.

- Economic Concerns: Profit, energy costs, waste treatment costs etc.
- Environmental Concerns: Atmospheric acidification, global warming, environmental burdens, ozone depletion, photochemical smog, resource usage etc.
- Health and Safety Impact: Health and safety risks such as risk of exposure, explosion, flammability etc.

Table 3.2-breakdown of all metrics analyzed by the SUSTAINABILITY EVALUATOR (Shadiya, 2010)

Economic Index	**Environmental Index**		**Social Index**	
	Environmental Burden	**Resource Usage**	**Total Inherent Safety Index**	**Health Risks**
Product Revenue	Acidification	E-Factor	Heat of Reaction	Carcinogenic Risk
Energy Costs	Global Warming	Mass Productivity	Flammability Index	Immune System Damage
Raw Material Costs	Ozone Depletion	Reaction Mass Efficiency	Toxic Exposure Index	Skeletal System Damage
Annualized Capital Costs	Smog	Energy Intensity	Explosivity Index	Developmental Damage
Waste Treatment Costs	Ecotoxicity to Aquatic Life	Water Intensity	Temperature Index	Reproductive System Damage
Material Value added	Eutrophication		Pressure Index	Respiratory System Damage
Profit	Aquatic Oxygen Demand		Corrosivity Index	Cardiovascular System Damage
			Equipment Safety Index	Endocrine System Damage
			Safety Level of Process Structure	Liver Damage
				Nervous System Damage
				Sensory System Damage
				Kidney Damage

Economy in every industry is the most regarded aspect and the main incentive continual operation of industries. One of the main goals of every industry is to minimize costs and maximize the profit. Also economy is one of the main aspects of sustainability and a process which is not economically viable cannot be sustainable. However the sustainability evaluator tool

presented by Shadiya (2010) is not independent in terms of calculating economic metrics, as the cost information has to be calculated using other available tools and then inserted to the tool. For this reason in this work the costs associated with the process are calculated based on the existing cost estimation methods presented in details in later chapters. The set of metrics for economic aspect which are calculated using the cost information are explained in subsequent sections of this chapter. Environmental aspect of sustainability is addressed by using a set of specific metrics and the social aspect is evaluated based on safety risk and health impact assessment (Shadiya 2010).

The advantage of using the metrics presented in this tool is that as summarized in previous section, no other sustainability assessment tool is as comprehensive as this one in providing a collection of metrics which cover all metrics of the sustainability. Nevertheless there are some other metrics for each aspect which need to be added in order to complete sustainability evaluation.

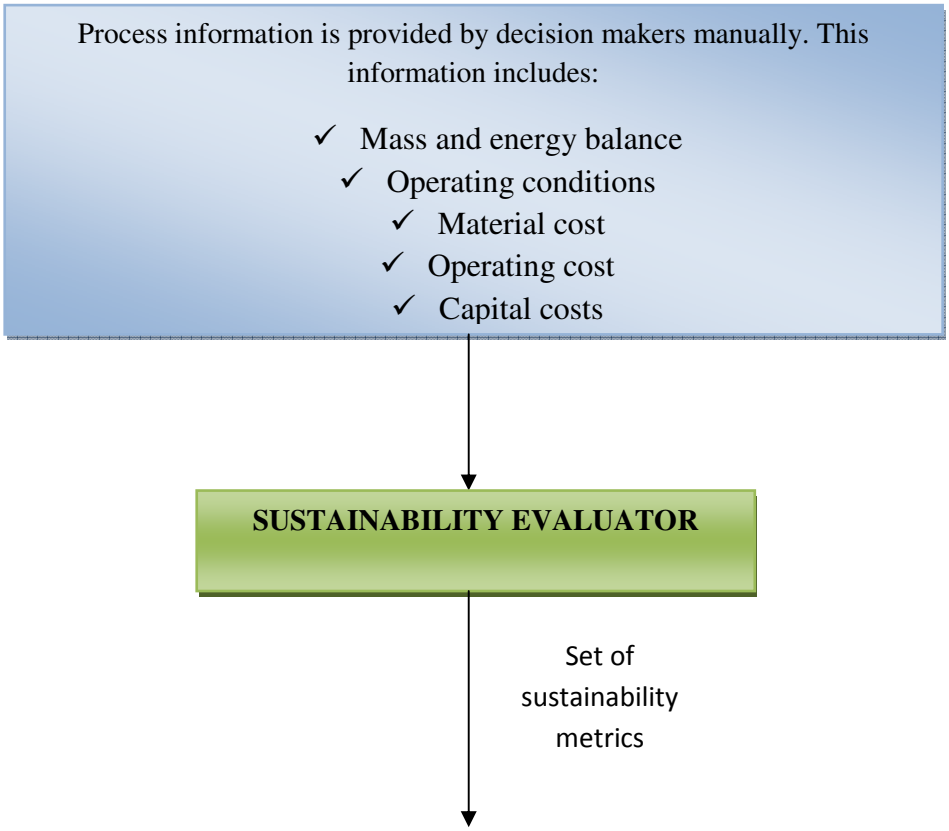


Figure 3.2-Sustainability Evaluator structure

The metrics presented in this tool are used for evaluation of process alternatives in terms of economic, environmental, health and safety concerns in the proposed framework. The concepts and calculation method for these metrics are presented in the following sections.

3.2.2 Economic Impact Metric

The economy assessment of a process is one of the most important tasks of process design and it is also one of the three main aspects of sustainability. There are several methods for assessing the economy of a process which some of them are presented by Dantus (1999), Seider et al. (2008) and Turton et al. (2009). In this work the set of economic metrics which are used for economic evaluation are listed below:

- Product Revenue: the product revenue is the measure of the total income produced by selling main products and by-products

- Raw Material Costs: this is the end cost of buying and transporting the raw material into the process boundary

- Waste Treatment Costs: all the costs imposed for treating and disposing wastes are presented by this metric

- Operating costs: the cost of all utilities consumed in the process is considered as operating cost
- Material Value Added: the difference between price of products and raw material are defined as this term (Carvalho et al., 2008).

- Annualized Capital Costs: this is the total fixed capital cost and working capital cost multiplied by capital recovery factor, CRF.

$$CRF = i(1+i)^n/(1+i)^n-1 \quad (3.1)$$

Where

CRF = Capital Recovery Factor

n = Number of Years

i = Interest Rate

- Profit: the profit is defined as the annual revenue minus the all annualized costs.

Profit = Product Revenue + By-product Revenue - (Raw Material Cost + Waste Treatment Cost + Operating Cost +Annualized Capital Cost) (3.2)

3.2.3 Environmental Burden Metrics

Another important aspect of sustainability is the environmental aspect which includes two categories: environmental burden and, resource usage. Chemical Engineers (IChemE Metrics, 2002), Green Metrics (Constable et al., 2002) and Bridges to Sustainability (Tanzil and Beloff, 2006) presented metrics and indices for assessment of the environmental impacts. The metrics suggested for this purpose are: global warming, stratospheric ozone depletion, photochemical smog, aquatic oxygen demand, atmospheric acidification, aquatic acidification, eco-toxicity to aquatic life, eutrophication for environmental burden and E-factor, reaction mass efficiency, mass productivity, mass intensity, energy intensity and water intensity for resource usage (Shadiya 2010). Description and calculation of mentioned sustainability metrics introduced by Shadiya (2010) which are used in this PhD work are presented in the following sections.

3.2.3.1 Global Warming

Global warming is the increase in the temperature of the earth surface caused by industrial and human activities. Some chemicals facilitate the global warming but carbon monoxide has the most contribution. Therefore, other components which cause global warming are converted to carbon dioxide using potency factors (Shadiya, 2010).

The potency factor of each component is multiplied to its total amount released to the environment and the result is added up to the value calculated for the other components existing in the process to assess the global warming. The potency factor of components which are used for calculation of global warming metric are presented in Appendix A Table 1.

3.2.3.2 Stratospheric Ozone Depletion

Ultraviolet rays are causing different skin diseases nowadays. As the ozone layer is depleted due to chemical components released to the atmosphere the amount of ultraviolet rays reaching the surface of the earth is being increased. Thus the ozone layer is playing an important role and its depletion should be addressed by sustainability environmental aspect. Several chemicals can cause the ozone layer depletion. Likewise the previous metric for this metric the listed chemical are converted to trichlorofluoromethane equivalent by multiplying their mass amount emitted to the atmosphere by the potency factors (Shadiya,2010). The chemicals causing ozone depletion and their corresponding potency factors are listed in Appendix A Table 2.

3.2.3.3 Photochemical Smog Formation

The reaction which produces smog at the right temperature and in presence of sunlight from petrochemicals and combustive chemical components is called photochemical smog formation (IChemE Metrics, 2002).For calculating this metric all chemicals which causesphotochemical (smog) formation are converted to ethylene equivalents by their corresponding potency factors shown in Appendix A, Table 3.

3.2.3.4 Ecotoxicity to Aquatic Life

Eco-toxicity to aquatic organisms is indicated by this metric. Eco-toxicity to aquatic organisms is caused by the presence of contaminants in water sources (Shadiya,2010). All substances are converted to copper equivalent using the potency factors shown in Appendix A, Table 4.

3.2.3.5 Aquatic Oxygen Demand

This metric indicates the increase in oxygen demand by aerobic bacteria because of contaminants in the water sources. All chemicals causing this problem are converted to oxygen equivalent by multiplying to potency factors which are shown in Appendix A Table 5 (Shadiya, 2010).

3.2.3.6 Atmospheric Acidification

Discharging Ammunia, sulfuric acid, hydrochloric acid, hydrogen fluoride, nitrogen dioxide and sulfur dioxide increase acidity in environment (IChemE Metrics, 2002; da Costa and Pagan, 2006). This is measured by Atmospheric Acidification metric. This metric is calculated by multiplying the mass flow rate and potency factor of each chemical. The chemicals causing atmospheric acidification are presented in Appendix A Table 6 (Shadiya, 2010).

3.2.3.7 Eutrophication

Nutrients may be discharged to water sources unwillingly which may cause undesirable plant growth. This metric is calculated by converting all chemicals to phosphorus equivalent using potency factors. The potency factors used for this metric are shown in A Table 7 (Shadiya, 2010).

3.2.3.8 Resources Usage

Resource usage consists of several metrics that each of them addresses different type of resource usage. The reader is referred to the works by Constable et al. (2002) and Tanzil and Beloff (2006) for more information. The sub-metrics of this category are listed below with their calculation method (Constable et al., 2002; Tanzil and Beloff, 2006).

$$E\text{-}Factor = Total\ Waste\ /\ mass\ of\ Product \tag{3.3}$$

$$Reaction\ Mass\ Efficiency = Mass\ of\ Product\ /\ Mass\ of\ Reactants \tag{3.4}$$

$$Mass\ Productivity = (1\ /\ Mass\ Intensity) \times 100 \tag{3.5}$$

$$Mass\ Intensity = Total\ Mass\ used\ in\ a\ Process\ Step\ /\ Mass\ of\ the\ Products \tag{3.6}$$

$$Energy\ Intensity = Energy\ Consumed\ /\ Mass\ of\ Product \tag{3.7}$$

$$Water\ Intensity = Water\ Consumed\ /\ Mass\ of\ Product \tag{3.8}$$

Except for reaction mass efficiency and mass productivity for all other environmental metrics the lower the value of metrics the more sustainable is the process.

3.2.4 Social Metrics

Social aspect of sustainability includes a variety of societal impacts. Many social metrics have been developed that each indicates a different impact on societies. For the scope of this work health and safety metrics of social aspect are selected. Due to importance of safety ion process design and also health impact of processes these metrics have been studied by many researchers such as Heikkila, (1999) and Tugnoli et al., (2008b). They presented quantitative metrics for social impacts. The metrics developed by Heikkila (1999) for safety risk evaluation and metrics developed by Cancer (2009) and Card (2005) for health risk evaluation are usedfor the purpose of this work. These two categories of social impacts, Process Safety Risks and Health Risks, are explained in the following sections.

3.2.4.1 Process Safety Risk

Safety metrics used for this work include heat of main and side reaction index, flammability index, explosivity index, corrosive index, toxic exposure index, temperature index, pressure index, equipment process safety index and process safety structure index (Shadiya, 2010).

3.2.4.1.1 Heat of Main and Side Reaction Index

Reactions are the heart of chemical processes. The reactors that handle reactions with high temperature or high quantities of heat are potential source of dangers and need to be well designed and controlled. Also many process need to handle several reactions and side reactions. Thus, this metric addresses both main and side reactions (Shadiya,2010). This index is calculated using the following equation (Heikkila, 1999; Jensen et al., 2003). As Shadiya (2010) suggested the values calculated by this equation are scored between 0 and 8 and the 8 is corresponding to worst scenario cases. Appendix A, Table 8 presents the scores for this index.

$$\Delta H_r = products\ (H_f)_{products} - reactants\ (H_f)_{reactants} \quad (3.8)$$

3.2.4.1.2 Flammability Index

This index measures the possibility of flammable chemicals to start burning in the case of leakage in the process. This index is calculated based on flash point temperature of chemicals (Shadiya, 2010). The data used for calculation of this index in this work were obtained from "Chemical Process Safety" (Crowl and Louvar, 1989) which presented by Shadiya (2010). Again the index scores for this index range between 0 to 8, and 8 is corresponding to the worst scenario. Appendix A Table 9 presents the scores for this index.

3.2.4.1.3 Explosivity Index

Some gases can form an explosive mixture in presence of the air. This index evaluates the potential for forming this type of gas mixtures. The explosivity index is calculated as the difference between the upper explosive limit (UEL) and the lower explosive limit (LEL) of chemicals. UEL and LEL for explosive chemicals were obtained from Crowl and Louvar (1989), material data safety sheets and Dow Fire and Explosive Hazard Classification (American Institute of Chemical Engineers (AIChE), 1994), and presented by Shadiya (2010). The scores for this index are shown in Appendix A Table 10, ranging between 0 and 8 as suggested by Shadiya (2010).

3.2.4.1.4 Corrosive Index

This index evaluates the likelihood of corrosion in the plant equipment caused by acids, acid anhydrides and bases. The corrosion in plant in addition to causing high maintenance cost also increases the danger of toxic exposure, fire and explosion due to leakages Shadiya (2010). This index is calculated based on the material which used for the construction of the plant and they are listed in Appendix A Table 11. As suggested by Shadiya (2010) the scores for this index are between 0 and 4 and 4 is corresponding to the worst case.

3.2.4.1.5 Temperature Index

This index evaluates the potential risk in processes caused by the range of temperature in the process. This is very important because some material of construction in process fail due to cryogenic or high temperaturesShadiya (2010). The score for this index are listed in Appendix A Table 12 and the worst case has a score of 4 as suggested by Shadiya (2010).

3.2.4.1.6 Pressure Index

Same as previous index this index evaluates the potential risk caused by the range of pressure in the process. High pressure conditions need more control and can cause leakage and affect strength of process equipment (Heikkila, 1999). The score range suggested by Shadiya (2010) is listed in Appendix A Table 13.

3.2.4.1.7 Equipment Process Safety Index

Some equipment based on the type of operation which is carried out in them imposes higher level of risk to the process. For example, furnaces and fire heaters impose more risk to the process due to their high temperature and operation (Shadiya, 2010). This index evaluates the risk associated with the process due to the presence of these types of equipment. Appendix A Table 14 presents the range of scores for this index as suggested by Shadiya (2010).

3.2.4.1.8 Process Safety Structure Index

The reliability of process structure and design are evaluated using this index based on recommendations in engineering standards, design manuals and related incidents. The score range of this index is presented in Appendix A Table 15 (Shadiya, 2010).

3.2.4.1.9 Toxic Exposure Index

This index evaluates the health risk imposed by presence of specific chemicals in processes. The level of the risk is calculated using threshold limit value (TLV). The lower the TLV the more hazardous are the chemicals. The data for TLV are presented by the American Conference of Governmental Industrial Hygienists (2009). The score range for this index is shown in Appendix A Table 16. Shadiya (2010) suggests the index score ranges between 0 and 65 whereas, 65 is for the worse safety scenario.

3.2.4.1.10 Summary of Safety Metrics

Heikkila (1999) proposes to use a cumulative safety index which is calculated as the sum of all mentioned safety indices. The range of safety score for chemical processes is listed in Appendix A Table 17. A chemical process with a score of 100 is considered as an extremely unsafe process (Shadiya,2010).

3.2.4.2 Health Risk

In order to address the health risk associated with processes the following indices are calculated: carcinogenic health risk, developmental health risk, reproductive health risk, cardiovascular health risk, endocrine system health risk, liver damage health risk, immune system damage health risk, kidney damage health risk, skeletal system damage health risk, neurological damage health risk and respiratory system health risk(Shadiya,2010).

3.2.4.2.1 Carcinogenic Health Index

This index evaluates the carcinogenic risk of processes based on the chemicals which are present in the process. International Agency for Research on Cancer categorizes the chemicals to four groups: carcinogenic to humans, probably carcinogenic to humans, possibly carcinogenic to humans, carcinogenic to animals but not humans and probably not carcinogenic to humans. In order to calculate this index the carcinogenic score of each chemical is multiplied by its being emitted to the environment. The scores and their corresponding groups are listed in Appendix A Table 18 (Shadiya, 2010).

3.2.4.2.2 Developmental Health Risk

This index calculates the risk of developing an unhealthy child by a woman exposed to toxicant chemicals. For this work the list of toxicant chemicals and their associated scores were

obtained for the work by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score (Shadiya, 2010).

3.2.4.2.3 Reproductive Health Risk

Risk of all type of damages to the reproductive system of adults who are exposed to reproductive toxicants is evaluated using this index. The possible reproductive damages can be listed as: abnormal sexual behavior, decreases in fertility, loss of the fetus during pregnancy (Shadiya, 2010). The chemicals suspected as reproductive toxicants were obtained from Shadiya (2010).

3.2.4.2.4 Circulatory System Damage Health Risk

The risk imposed due to exposure to cardiovascular toxicants to the circulatory system of adults is indicated by this index. Arteriosclerosis, cardiac arrhythmia, and decreased coronary ischemia can be considered as the circulatory system problems (Shadiya, 2010). The chemicals suspected as cardiovascular toxicants were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010).

3.2.4.2.5 Endocrine System Damage Health Risk

Risk imposed to the endocrine system of an adult body is measured by this index. Several endocrine problems which can be caused by exposure to chemicals categorized as the endocrine toxicants are: hypothyroidism, diabetes mellitus, hypoglycemia, reproductive disorders, and cancerShadiya (2010). The chemicals suspected as endocrine toxicants were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010).

3.2.4.2.6 Gastrointestinal and Liver Damage Health Index

The risk imposed to gastrointestinal tract, liver, or gall bladder of adults due to contact to chemical toxicants. The chemicals suspected as endocrine toxicants were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.7 Immune System Damage Health Risk

The risk of being infected by many diseases is increased due to damages to the immune system of body. The immune system can be weakened or damaged by exposure to specific chemical toxicants (Shadiya,2010). The chemicals suspected as immunotoxicants were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.8 Kidney Damage Health Risk

Risk of damages to the kidney, ureter and bladder of adults due to exposure to chemicals is evaluated by this index. The suspected were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.9 Skeletal System Damage Health Risk

The risk imposed to the bones, muscles and joints by exposure to the toxic chemicals is evaluated by this index. Skeletal system damage including arthritis, fluorosis and osteomalacia can be caused by chemical toxicants (Shadiya, 2010). The suspected chemicals were obtained

from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.10 Nervous System Damage Health Risk

The risk to the nervous system including confusion, fatigue, irritability, brain damage and loss of coordination due to exposure to specific toxicants is measured by this index. The suspected chemicals were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.11 Respiratory System Damage Health Risk

The risk of nasal passages, pharynx, trachea, bronchi, and lungs of adults being damaged due to exposure to chemical toxicants is evaluated by this index. The common respiratory system damages are: acute and pulmonary edema, irritation, bronchitis irritations, emphysema, and cancer (Shadiya,2010). The suspected chemicals were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.12 Skin or Sensory Organ Damage Health Risk

The risk of the skin or sensory organ damages is due to toxic exposure is measured by this index. The damages of sensory organs include hearing loses, sense of smell, eye irritations etc. The suspected chemicals were obtained from Shadiya (2010) and the score for all these chemicals are assigned 0.6 as suggested by Shadiya (2010). To calculate the index for each chemical, its amount emitted to the environment is multiplied by its score.

3.2.4.2.13 Summary of Health Metrics

The scoring ranges for the twelve health indices presented in previous sections are listed in Appendix A Table 19. To calculate the indices for each chemical, its amount emitted to the environment is multiplied by its corresponding score. The list of toxic chemicals for each index can be found in Shadiya (2010).

3.3 Incorporating the Sustainability Metrics to the Proposed Framework

All the chemicals required for calculating the sustainability metrics mentioned in previous sections along with their corresponding scores and potency factors were stored in a form of MATLAB structure to be accessible during the optimization procedure of the framework. As shown in Figure 3.3 to 3.6 using interactive interfaces, process designers are asked to select chemicals existing in the process for each sustainable category and metric.

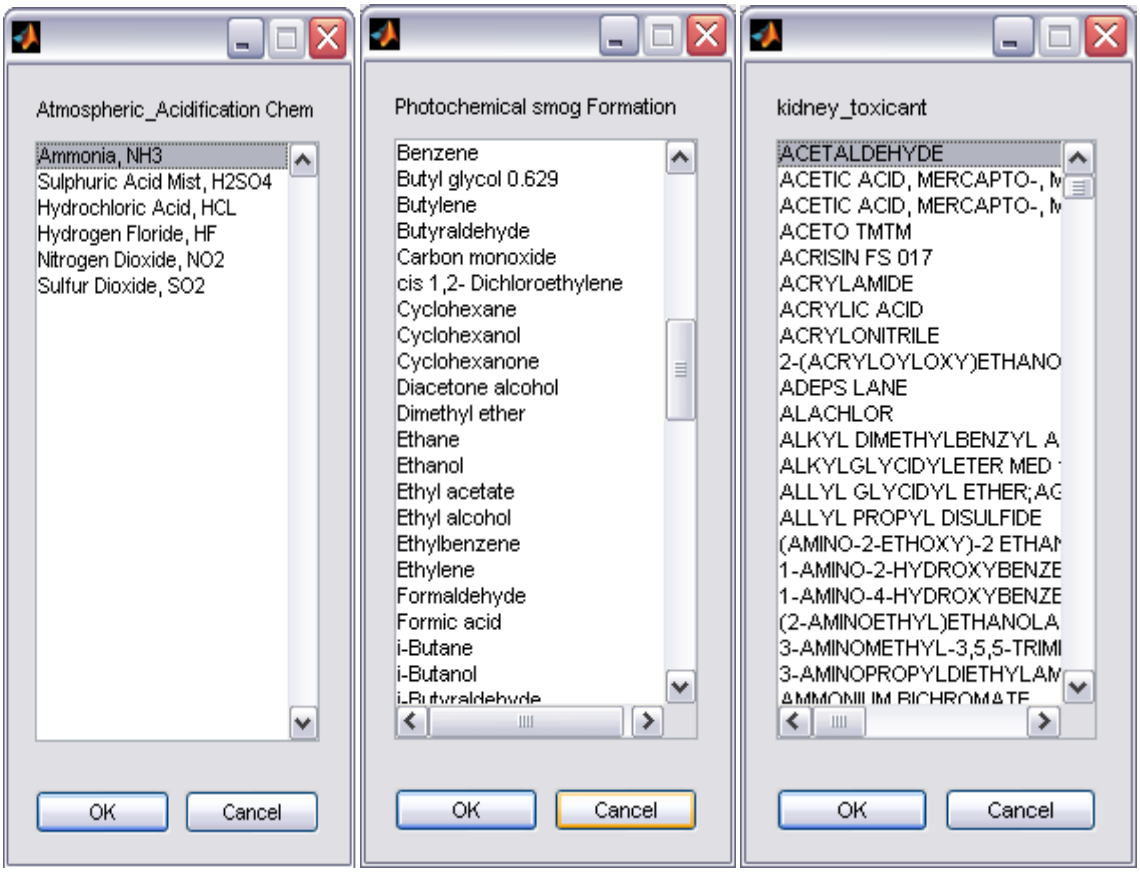


Figure 3.3-examples of the framework interface for inquiring information for sustainability metrics calculation

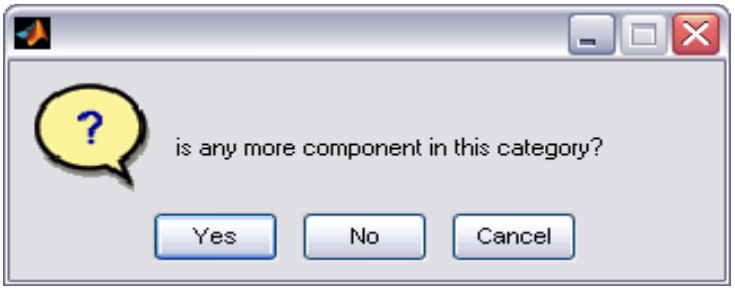


Figure 3.4-message to ensure completion of information in each category

After popping each up category a message is shown to the user as shown in Figure 3.4 to ensure that all chemicals in that particular category which are present in the process have been selected.

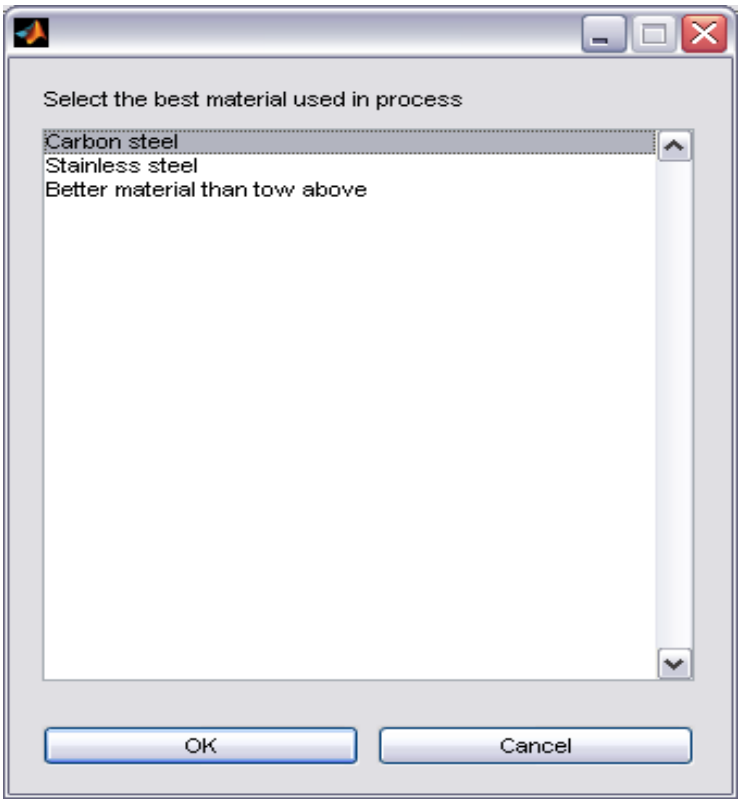


Figure 3.5-inquiring the construction material of the process

Material of construction in the process is selected using the above interface of the framework. And this information is used as explained before for calculating the safety metric.

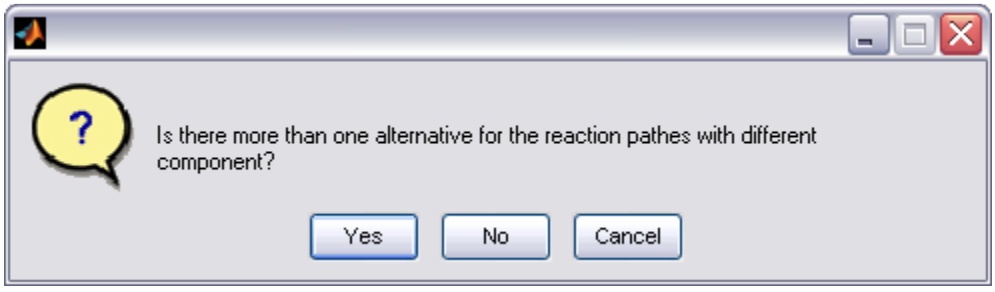


Figure 3.5-inquiring the construction material of the process

The message shown in Figure 3.6 is shown to the decision maker in order to get information about the reaction path alternatives. More than one reaction pathway alternatives require more sustainability metrics to compare the sustainability of process. If user chooses “Yes” for this

message, reaction mass efficiency also could be considered as an objective for design. Similar messages are shown to the user for inquiring about alternative chemicals which can be used in the process. The formulation of the optimization problem can vary if there is more than one option for the combination of chemical components in the process. Different combinations of chemical components may be considered for example in the cases of having more than one reaction pathway or several options for solvents in separation units.

Once all the chemicals are selected and the information about the structure and configuration of the process is obtained using the interactive sustainability interface of the framework, a matrix is built by the program to restore all the information in memory of the computer to be used later at each iteration of the program. The interaction of the sustainability module of the framework with other modules especially with the optimization module is explained in later chapters of this dissertation.

3.4 Summary

In this chapter a brief review of existing method for sustainability aspects was provided. Also the methodology for evaluating and comparing the process alternatives in terms of three aspects of sustainability was explained. All three aspects of sustainability are evaluated using several quantitative metrics which are all calculated systematically using a programmed sustainability module in MATALAB. This module uses the mass and energy information of process alternatives, produced by the optimization algorithm and process simulator, to evaluate them in terms of sustainability dynamically.

CHAPTER 4

MULTI-OBJECTIVE OPTIMIZATION IN CHEMICAL ENGINEERING: METHODS AND APPLICATIONS

4.1 Process Optimization and Background

Process optimization is an essential need for systematic process synthesis and conceptual design and it leads to better design based on sustainability criteria. As elaborated in previous chapters, most of sustainability criteria are in conflict with traditional process design criteria and in some cases with other sustainability criteria. This would justify the use of multi-objective optimization (MOO) algorithms for process design and decision making. Many researchers and academicians have written several books on the methods and applications of optimization such as Lapidus and Luus (1967), Beveridge and Schechter (1970), Himmelblau (1972), Ray and Szekely (1973), Floudas (1995 and 1999), Luus (2000) and Edgar et al.(2001), Tawarmalani and Sahinidis (2002), Diwekar (2003), and Reklaitis et al .(2006). This chapter reviews the basic concepts of optimization and optimization algorithms as well as their application in chemical engineering.

Most of the mentioned books and optimization applications have been limited to the optimization of only one objective at a time, called single objective optimization (SOO). However, real life applications and the recent trend in chemicals design requires considering more than one objective simultaneously for process design. These objectives could be different metrics of sustainability or different facets of the economic concern in process design such as

capital cost, operating cost, profit, payback period, selectivity of reactions, production, recovery of the product, reaction conversion, energy consumption, process efficiency, process safety, controllability, process flexibility, operation time, etc. (Rangaiah, 2009). Often, the objective functions of a process optimization problem are conflicting which means the decision variables which make one objective optimum do not correspond to an optimum of the other objectives or in some cases worsen the value of the other objectives. A traditional method for handling more than one objective is to combine the objectives as a single objective. These methods are mentioned in later sections of this chapter.

On the other hand the multi-objective optimization method, also called multi-criteria optimization, is a more suitable approach for handling multiple of conflicting objectives. As compared to single objective optimization where the final solution is limited to one optimum solution, except when the objectives are not conflicting, the solution of a multi-objective optimization usually consists of a set of optimum solutions. Therefore, there are several approaches for the selection of a unique solution. Advance computational facilities and efficient methods for handling MOO problems have increased the possibility of using multi-objective approaches in chemical engineering decision making problems. As stated by (Rangaiah, 2009) more than 100 application of MOO were reported in 130 publications in chemical engineering from year 2000 to 2007. Some notable books and publications on the methods and applications of MOO in chemical engineering and outside the realm of chemical engineering have been presented by: Diwekar (2003), Cohon (1978), Hwang and Masud (1979), Chankong and Haimes (1983), Sawaragi et al. (1985), Stadler (1988), Haimes et al. (1990), Miettinen (1999), Deb (2001), Coello Coello(2000 a) and Tan et al. (2005).

4.2 Basic Concepts of Multiple Criteria Optimization

A general multi-objective problem consists of two or more conflicting objectives to be either maximized or minimized with several decision variables and constraints. The mathematical

representation of the problem and constraints could be linear or nonlinear. A problem with linear constraints and objective functions is called a linear optimization problem. In this section to illustrate the basic concepts of multi-objective optimization a minimization problem with two conflicting objective is considered. A general representation of a problem with two objectives is as following:

$$Minimize \{f_1(x,y), f_2(x,y), \ldots, f_n(x,y)\} \quad (4.1)$$

$$\{x,y\}$$

Subject to

$$h(x,y) = 0 \quad (4.2)$$

$$g(x,y) \leq 0 \quad (4.3)$$

$$x_L \leq x \leq x_U \quad and\ y \in \{0,1\} \quad (4.4)$$

Where x is the decision making vector, $h(x,y)$ is the vector of equality constraints and $g(x,y)$ is the vector of inequality constraints.

It should be mentioned that maximization and minimization problems can be easily converted together using transformations for monotonic increasing and decreasing functions. The most common transformation method is to multiply the objective by -1. Hence, the generality of the concepts explained in this section is preserved.

The decision variables can include integer or binary variables. In this case the problem would be a mixed integer optimization problem. In the realm of chemical engineering the binary variables could appear in the mathematical formulation of the problem to indicate process configuration choices and mixed integer variables could be used to represent the number of stages for a specific operation. Usually equality constraints are used to preserve the fundamental mass and energy balances in chemical engineering and inequality constraints enforce limitations on the process such as production limits or operational conditions restrictions. Thus, the equality constraints and active inequality constraints divide the multi-dimensional search space into feasible and infeasible regions. The feasible region is a segment of the whole search space and all

decision vectors belong to that region; satisfy all equality constraints and active inequality constraints including variable bounds. Most often, problems in chemical engineering are nonlinear due to inherent nonlinearity of models and equations in chemical engineering. Also, very often most of nonlinear problems are nonconvex and have multiple local minima. For further information on the concept of convexity the reader is referred to "Convex Analysis and Non Linear Optimization" by Borwein and Lewis (2006).

Due to conflicts between objectives of MOO problems many solutions exist for a problem presented in Equation 4.1. The collection of all final solutions of a MOO problem is called the final solution set. None of the members of the final solution set is inferior to the other members as each member is better than other members at least in terms of one optimization objective. This occurs due to the fact that when one objective improves at least another objective worsens. This is what is referred as conflict between the objectives. The solution set of MOO problems is called Pareto solution in reference to the contribution of Edgeworth and Pareto, two economists in 19th century. Also there are other common names for the final solution set of MOO problems based on the characteristics of the solutions such as non-dominated or simply Pareto solutions (Rangaiah, 2009).

Definition 1:

A decision vector x is called non-dominated (Pareto solution) if and only if there is no other decision vector, x', such that all $f_i(x') \leq f_i(x)$.

Comparison and discussions on the solutions of multi-objective problems and also the performance of MOO algorithms usually are done based on objective functions space. However, the optimization procedure is performed in the decision space and solution vectors are varied and improved based on their relative characteristics to other vectors in the decision making space. The reader is referred to the work by Tarafder et al. (2007) for further information on finding multiple solution sets in MOO of chemical processes.

In Figure 4.1, the blue dots represent Pareto solutions which are not inferior to any other solution and the green dots represents the feasible points but inferior with respect to the Pareto frontier (the convex continuous line).

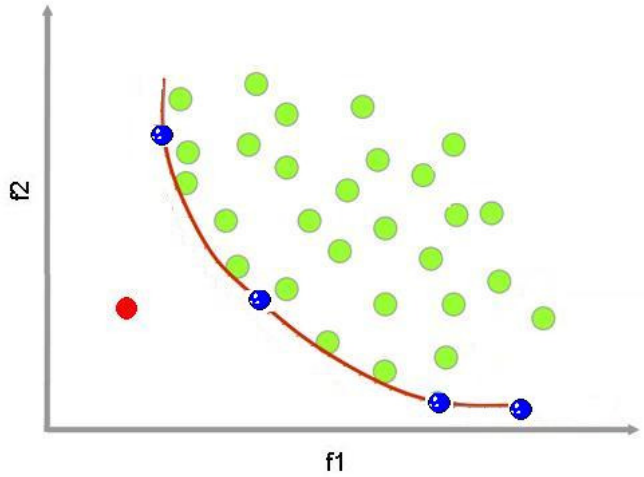


Figure 4.1- An Example of a Pareto Optimal Solution Set

4.3 Multi-Objective Optimization Strategies

There are various types of optimization algorithms for handling MOO problems. They have been classified by several researchers. One classification based on the number of solutions generated by the algorithm was presented by Miettinen (1999) and Diwekar (2003). They classify the multi-objective optimization methods into preference-based and generating methods. Preference based methods quantify the decision-makers' preference and the solution that best satisfies the decision-makers' choices is then selected. Generating methods have been developed to find the exact Pareto set or an approximation and they find a set of solutions for the problem instead of only one preferred solution. Figure 4.2 represents the overall classification of MOO algorithms.

No-preference methods include global criterion and the multi-objective proximal bundle method which are not dependent on any preference and can find solutions close to the ideal

objective vector. Posteriori methods include several methods which will be explained in more details later in this chapter. On the other hand, preference based methods include two main sub categories: priori methods and interactive methods. Interactive methods are based on preferences of the decision maker which are obtained interactively at specific iterations of the algorithm. After each iteration the decision maker uses the information from the solutions to rectify the preferences. The modified preferences are used in the next iterations of the algorithm to find the final solution. Interactive surrogate worth trade-off methods are the most common examples of this type of optimization methods (Rangaiah, 2009). Priori methods are explained in the next section.

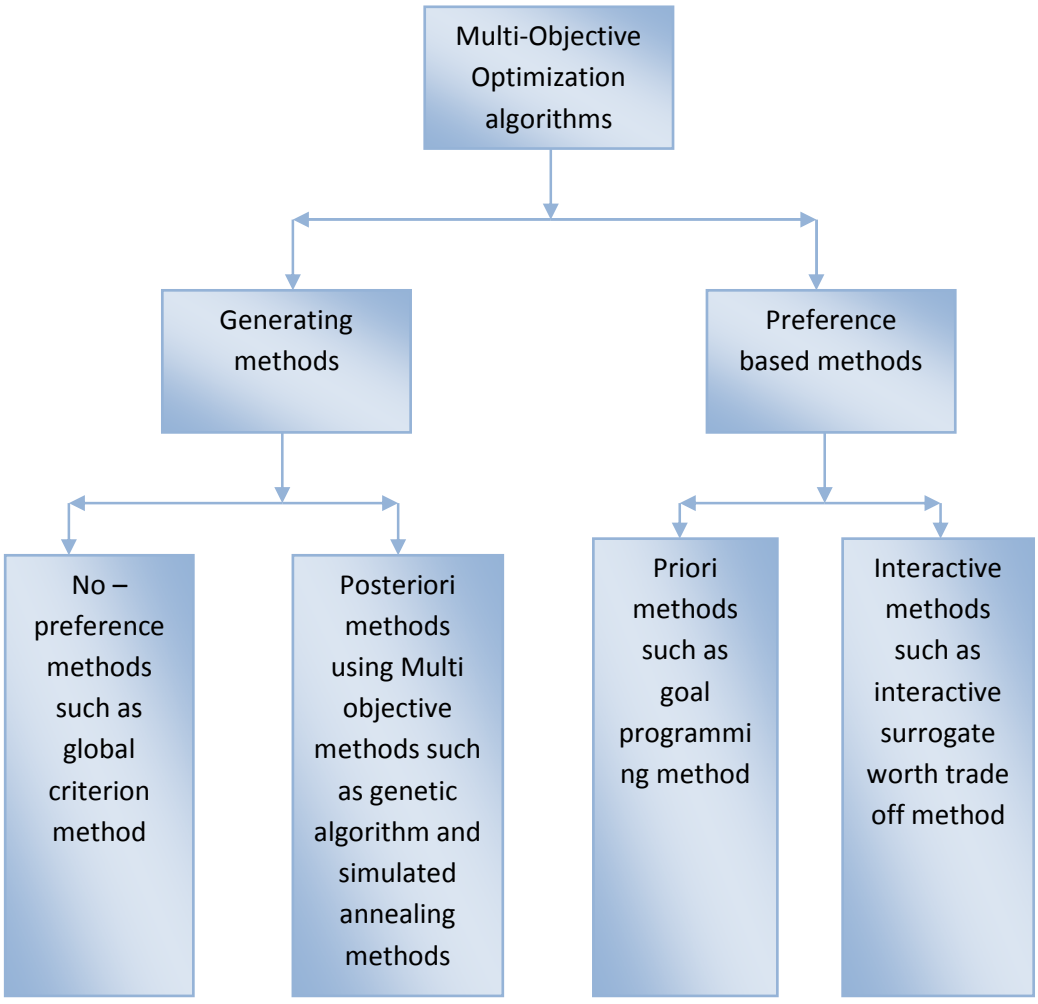


Figure 4.2- classification of MOO algorithms (Rangaiah, 2009)

4.3.1 Priori Methods

In general these methods transform multi-objective problems to a single objective problem which can be solved by single objective optimization methods. There exists a variety of single objective optimization methods which can be found in books written by Lapidus and Luus (1967), Beveridge and Schechter (1970), Himmelblau (1972), Ray and Szekely(1973), Floudas (1995 and 1999), Luus (2000) and Edgar et al.(2001), Tawarmalani and Sahinidis (2002), Diwekar (2003), Reklaitis et al .(2006). However, the review of these methods is out of scope of this dissertation.

The transformation of multi-objective optimization problem into a single objective problem is done with several ways which are presented in detail by (Ehrgott, 2005; Marler and Arora, 2004). MOO problems could be transformed to a SOO problem by normalization methods which are dependent on the decision maker's preferences. The preferences are required most of the time prior to starting the optimization procedure. However, as these methods are dependent on the preferences of the decision maker to normalize the objective functions, often there are complexities with obtaining valid preferences to normalize the objectives. Valid preferences require priori knowledge about the problem and behavior of the objective functions and sometimes is almost impossible due to complete different nature of the objectives. The next sections provide a brief description of the most common Priori optimization methods. However, in cases that an ideal solution for a multi-objective problem is given, application of satifaying SOO methods such as goal programming could reduce computational burdon of finding a satisfaying solution.

4.3.2 Generating Methods

As mentioned in the previous section, one common limitation among all the preference-based methods is that it is difficult and complicated for the decision-maker to provide and formulate

consistent preferences. Also they are not well suited for handling multi-objective optimization problems as they do not result in well distributed Pareto solutions and their implementation for MOO problems is often computationally expensive. Generating methods overcome this limitation by providing a non-dominated optimal solution set independent of preferences allowing the decision-maker to select the final solution as a tradeoff between objectives (Marler and Arora; 2004; Miettinen, 1999). Posteriori methods generate a Pareto solution with finite number of solutions which enables the decision maker to decide for the final solution. The number of Pareto solutions is a controllable parameter in these methods and could be increased to cover the true Pareto front with well distributed solutions.

Being capable of generating diverse final Pareto solutions along the true Pareto front covering the extremes of the objective functions is an important feature of posteriori generating methods. Often these methods are criticized due to their high computational expense and also the cognition load that is imposed by the generated Pareto solution set, which still consist of many non-inferior solutions, on the decision maker (Shin and Ravindran, 1991).

Traditionally MOO problems were handled by single objective optimization methods by consistently solving a sequence of SOO problems. To do so the multi-objective problems are reformulated as a normalized single objective function. These methods were mentioned in the previous sections. However, most of the traditional methods which apply a single objective algorithm for generating a Pareto solution for MOO problems fail to generate a diverse solution set and they are incapable of locating Pareto solutions in nonconvex regions. Thus, a new generation of optimization algorithms has been proposed by researchers in this field. Athough they are better suited for handling multi-objective problems, they are essentially capable of handling single objective optimization problems as well. The main goals of generating optimization methods are to:

- Minimize the distance between the true Pareto front and the Pareto solutions obtained
- Obtain a well distributed solutions along the true Pareto front

- Maximize the spread of the final Pareto solutions to include the extremes of the optimization objectives

Before proceeding to the next sections to talk about the generating methods, another classification of optimization algorithm is explained here. This helps to better understand the classification of generating methods. Table 4.1 summarizes multi-objective optimization methods.

Table 4.1 – Feature and Limitation of Different Categories of Multi-Objective Optimization Methods (Rangaiah, 2009)

Optimization methodology	Features and limitations
No Preference Methods (e.g., global criterion and neutral compromise solution)	These methods do not employ any preference from the decision maker either before, during or after solving the problem. However, an ideal vector of objective functions must be provided.
Methods Using Scalarization Approach	These classical methods require solution of SOO problems many times to find several Pareto-optimal solutions. e-constraint method is simple but ineffective for multi-objective problems. Also determining the boundaries on objective functions encounters difficulties. Weighting method fails to find Pareto solutions in the non-convex region and selecting suitable normalization factors is the source of difficulties for this method.
Posteriori Methods Using Multi-Objective generating approach	These relatively recent methods have found many applications in chemical engineering. They generate a set of Pareto-optimal solutions and thus provide cover the entire Pareto front. The decision maker is only involved after finding the Parteo solution set to select one of the non-dominated solutions based on s/he preferences.
Priori Methods (e.g., value function, lexicographic and goal programming methods)	These methods require preferences from the decision maker in advance. This causes a lot of difficulties in finding appropriate preferences with no/limited knowledge on the functionality of objective functions. Also they only one optimum solution based on the provided preferences.
Interactive Methods (e.g., interactive surrogate worth tradeoff)	Decision maker interactively provides preferences during the optimization procedure. This method can find more than one objective according the DM's preferences. However, this method

	requires preferences for the DM which are not easy to attain. Lots of time and effort is required for this method which may not be practical.

4.3.3 *Deterministic versus Heuristic Methods*

There are two major classes of optimization algorithms which are distinct from each other based on the mathematical theory behind the algorithm. Deterministic methods are referred to those methods which have an absolute mathematical theory and use gradients of objective functions and constraints to lead the algorithm to an exact solution. On the other hand, the heuristic methods which encompass a variety of methods do not have a universal definition. Nevertheless, methods which find acceptable solutions in a reasonable time but they lack a profound theoretical background are called heuristic methods. Heuristic methods are usually based on naturally inspired rules.

Due to the importance of global optimization and difficult optimization problems, the development of heuristic methods has been initiated by researchers. Some of these methods have been explored mathematically such as simulated annealing. On the other hand, some of the heuristic methods which proved to be very efficient in handling difficult optimization problems are too complex for mathematical investigation. Therefore, there islittle theory behind most of them such as evolutionary and population based methods Zhigljavsky and Zilinskas (2007). Stochastic methods are a main active sub category of the heuristic methods which also are called metaheuristic methods. These methods search the space using a random technique which also could be adaptive random search. The adaptive random search pushes the algorithm to search the feasible area in the entire solution space.

Both approaches have been used for industrial process optimization. Whereas preference based methods can be applied with deterministic methods, generating methods are better suited for heuristic methods (Grossmann and Guillen-Gosalbez, 2009). Many deterministic methods

have been used to address environmental criteria in processes focusing on a single objective approach. However, their application to multi-objective optimization is limited. The next sections of this chapter summarize the drawbacks of each class of optimization methods and give a short review of the methaheuristic methods.

4.3.3.1 Drawbacks of Deterministic Methods

Deterministic methods have been used to optimize optimization problems and many deterministic methods have been applied for superstructures optimization which is the subject of this work (Diwekaret al. 1992; Diwekar and Rubin, 2003; Grossmann and Guillen-Gosalbez, 2009; Grossmann and Kravanja, 1995; Ivakpour and Kasiri2009; Kamath et al., 2010; Lee and Grossmann, 200). Although this class of optimization methods are exact and very timely efficient in handling single objective problems, they have some limitations which initiate use of methaheuristic methods vastly. The most conspicuous limitations could be listed as following:

- Due to the fact that movement toward the optimum in the deterministic methods are based on gradient of the objective function, in nonconvex optimization problems which multi local optimum exist, these methods fail to locate the global optimum. Based on different initial guess used at the beginning of the optimization procedure these methods may trap in different local minima locations.
- Deterministic methods have been modified to search for the global optimum. There exist several algorithms which have been developed for global search which are discussed by different text books including Tuy (1998), Bard (1998), Sherali and Adams (1999), Horst et al. (2000), Floudas (2000a) Horst and Tuy (2003), Rubinov (2000), Strongin and Sergeyev (2000) , Tawarmalani and Sahinidis (2002a). However, the proposed global search methods which may sometimes justify use of deterministic algorithms are limited only to handle certain type of problems and they are generic.

- Since conventional techniques such as gradient based, simplex based algorithms and simulated annealing were not designed for multi-objective optimization, they are not very suitable to be extended to multi-objective problems (Costa and Oliveira, 2001).
- As mentioned before, methods such as the weighting approach, goal programming, and the e constraint method have to be utilized for handling multi-objective optimization problems using deterministic algorithms which are not suitable for large scale multi-objective optimization.
- In addition, deterministic algorithms are time consuming and erroneous in handling multi-objective problems. Surrogate models and pseudo variables have been introduced to circumvent the problem of implicit functionality (Diwekar and Rubin, 2011; Diwekar, 1992) However; surrogate models reduce the reliability of optimization.
- Use of deterministic methods is dependent on a priori knowledge of problem or simplifying of the objective functions.
- They cannot handle complex and implicit functionalities that appear in sequential modular software.

4.3.3.2 Metaheuristic or Stochastic Methods

Methaheuristic or stochastic optimization methods are heuristic methods which use adaptive random search to reach to the global optimum. Generally, methaheuristic methods explore the solution space using a sampling method and intensify the knowledge obtained from previous samples. There is a dynamic balance between local exploration and global search of the solution space using random sampling and incorporating the knowledge of the solution space adaptively (Blum and Roli, 2003). Some of the most successful and well performed stochastic algorithms are: Simulated Annealing, The Tabu Search, Genetic algorithm, Ant Colony, Particle Swarm Optimization and Shuffled Frog Leaping. One class of stochastic methods are based on a

population of randomly generated solution and adaptive rules to guide the population toward optimums of the objective functions. The most wildly spread methods of population based methods are Genetic Algorithm (GA), Differential Evolution (DE), Ant Colony and Particle Swarm Optimization (PSO)

Among heuristic methods, population based algorithms are very effective in locating global optimums. They simultaneously deal with a set of feasible solutions and an entire set of non-dominated optimal solutions in a single run while other methods must perform a series of iterations. These methods are not necessarily dependent on the gradient of functions for searching the space thus they are well suited for handling multi-objective problems. The main sources of effectiveness of methaheuristic methods are:

- Population based algorithms are also capable of handling complex functionality such as discontinuities, multimodality, disjoint feasible space and noisy function evaluations. They are less dependent on the shape and convexity of the objective function and constraints which is a big problem for deterministic algorithms.
- Population based algorithms enable better performance towards finding global optima in multimodal multi-objective problems (Costa and Oliveira, 2001).
- Population based algorithms can effectively handle the implicit functionalities.
- The structure of population based method allows them to handle multiple objectives simultaneously.
- These methods can handle MOO problems without using any prior knowledge of the problem and generate a Pareto solution set.

4.3.3.2.1 Multi-Objective Evolutionary Algorithms

Evolutionary algorithms are stochastic optimization techniques which were inspired by the natural evolution and Darwin's survival theory. Evolutionary algorithms have been started by

development of Genetic algorithms (GAs) (Holland, 1975; Goldberg, 1989), evolution strategies (ESs) (Rechenberg, 1965), and evolutionary programming (EP) (Fogel et al., 1966), in the 1960s and 1970s almost independently as the ancestors of the currently existing evolutionary algorithms. The majority of currently existing evolutionary algorithms are hybrid methods which their characteristics differ dramatically from their ancestors. Nowadays, Genetic Algorithm is the most well-known and most applied evolutionary algorithms. Deb (2001) and Goldberg (1989) mentioned the strengths of evolutionary algorithms comparing to the traditional methods. However, there are some deficiencies associated with these methods. Due the heuristic nature of these methods exact solutions cannot be guaranteed by using these methods. The solutions obtained are approximate solutions and may not be even acceptable in each run. Also the computational burden of these evolutionary methods is high because of exploiting a population of solution at each run.

4.3.3.2.2 Population Based Algorithms

Population based algorithms are a class of stochastic optimization methods which are very similar to the evolutionary algorithms. These algorithms employ a set of random or systematic samples of solution spaces to search for the solutions of optimization problems. However, solutions at each iteration of algorithms, the population members, evolve based on their social behavior and characteristics instead of their genetics. Two most common algorithms of this class are Particle Swarm Optimization (Kennedy and R. Eberhart, 1995) and Ant Clony algorithm (Dorigo et al., 1996).

Ant Colony (ACO) algorithm is one of the most popular population based algorithms. The social behavior of ants and the fact that ants can find the shortest distance between their nest and a source of food is simulated in this algorithm to find the optimal solution of a problem. However, this method is more suitable for single objective optimization.

Particle Swarm Optimization algorithm originally developed by Kennedy and Eberhart (1995) is a collection of particles, called the swarm, which cooperate as a population to search for the optimum solution of a problem. Each particle is a multi-dimensional array which its components (bits) are values of decision variables. Each particle carry two type of information with it to be used for calculation of its movement direction in the search space: first, the position of the best solution ever found by all the particles in the swarm (global best or gbest) and second; the best position ever found by the particle itself (local best or pbest). However, flying directly toward the pbest and gbest does not help exploring the search space because these points have been visited before. The main idea in PSO is to modify the speed of particles with incorporating inertia instead of directly changing the position. The speed of particles is modified using the information of pbest and gbest positions as well as their previous speed. The inertia component of the speed enables the particles to explore the entire search space. The structure of original particle swarm is as following:

1. Initiate the algorithm with a specified number of particles which were systematically or randomly were sampled from the search space.
2. Duplicate the best position of the best particle in the swarm and copy it as the gbest.
3. Set the current position of each particle as its pbest.
4. Calculate the new position of each particle based on Equation (4.5).

$$v^i_{j,,t+1} = w\, v^i_{j,t} + c_1 r_1 (G^i_{j,t} - x^i_{j,,t}) + c_2 r_2 (P^i_{j,t} - x^i_{j,,t}) \quad (4.5)$$

$$x^i_{j,t+1} = x^i_{j,,t} + v^i_{j,,t}$$

where:

t, is the iteration number, P and G respectively represent Personal Best and Global Best, w is the inertia constant, c_1 and c_2 are two positive constants and, r_1 and r_2 are two random values in between [0,1].

5. Evaluate each particle against objective function of the problem and find the gbest.
6. Repeat the steps 2 through 5 till stopping criteria is met.

The particles positions are updated based on three main components: first, the personal knowledge or cognitive component which attracts the particles to their own best position; second the social knowledge or the global knowledge which attracts the particles toward the best position ever found by the all particles in the swarm and third is inertia which is the contribution of the velocity of each particle in its new velocity vector. The last term in the updating equation helps random search and exploring new region in the search space. Also there should be a balance between social component and personal component of the updating equation. Simulations showed that too high a value for personal component relative to the social component results in perplexity of particles in the search space and a search more close to complete random search. On the other hand too high a value for social component relative to the personal component results in premature convergence. Kennedy and Eberhardt suggested approximately equal values, around 2, for the acceleration coefficients c1 and c2 to balance the movement of particles toward the gbest and personal best such that both global exploration and local exploration are preserved (Collet and Rennard, 2007).

4.3.4 Issues in Optimization of Chemical Engineering Problems

As explained in Chapter 2 of this dissertation, incorporation of optimization methodologies in different aspects of chemical engineering is a necessity. Chemical engineers and researchers have been using optimization algorithms for design and modification tasks since a long time ago. Works by Umeda et al. (1980), Clark and Westerberg (1983) are examples of application optimization methods in chemical engineering. However, application of multi-objective optimization methods was restricted in chemical engineering and nearly all optimization problems used to be solved by single objective optimization methods before 1980s. Bhaskar et al. (2000) shows that the most commonly used optimization techniques in chemical engineering for handling MOO problems were e-constraint, goal programming, and surrogate trade-offs. Since

late1990s the application MOO algorithms in chemical engineering appeared to be very useful and of significant importance.

However, the general application of multi-objective generating methods in chemical engineering, especially stochastic methods was limited comparing to other engineering disciplines such as electrical, industrial and mechanical engineering (Miettinen et al., 1999, Dasgupta and Michalewicz, 1997a). The main reasons contributing to this trend can be listed as following:

- Chemical processes often consist of several interconnected unit operations which all are modeled with highly nonlinear models. A rigorous model of a chemical process usually results in a very large scale highly nonlinear and nonconvex problem which is very hard to be solved (Biegler et al., 1997; Edgar and Himmelblau, 2001). Design problems can very complex such as a plant model reported 41147 variables, 37641 equality constraints, 212 inequality constraints, and 289 plant measurements (Lowery and coworkers 1993). For optimization problems with such a high level of complexity the computational capacity of the population based methods needs to be improved.
- The chemical engineering problems are usually highly constraints. The physical fundamentals of the process, mass, energy and momentum balances are ensured with equality constraints as well as the models which describe the behavior of the process. These equality constraints can be in form of algebraic, ordinary differential, and partial differential equations. In addition to that solving a set of highly nonlinear equations per se is numerically difficult (Dennis and Schnabel, 1996; Nocedal and Wright, 1999), presence of these equation in an optimization problem increases the number of equality constraints which are a source of complexity of problems to be handled by optimization algorithms. The presence of many nonlinear equality constraints in optimization problems could results in intricacies like rare, disjoint,

intricately scattered, or even empty feasible solution space. As the stochastic population based methods were not originally designed for handling constrained problems, there is no universal mechanism to handle constraints in optimization using these methods. Although several constraint handling techniques have been developed by researchers (Michalewicz, 1995a, Coello Coello, 2002) there is not a generic and robust technique suitable for handling all type of constrained optimization problems. The majority of test problems for stochastic methods exclude the equality constraints by relaxing them and converting them to inequality constraints. However, due to high strict need of satisfying physical constraints of chemical engineering models the extent that equality constraints can be loosened is extremely limited.

Considering the above discussion, it can be concluded that the existing stochastic optimization method need to be modified for handling large scale and highly constrained problems. The biggest challenges for these types of optimization methods to be applied in chemical engineering problems are: first incorporating a robust technique for handling constraints and second is to improve the computational capacity of algorithms and make the more timely efficient.

4.3.5 Constraint Handling

Real world optimization problems almost always are constrained problems. Thus, researchers have put forth effort to develop constraint handling method for stochastic based optimization approaches. Michalewicz (1995a), Michalewicz and Schoenauer (1996) and Ceollo (2002) have reviewed several constraint handling methods which have proposed during past decades. Due to presence of constraints the search space (decision space) of optimization problems is divided to two main regions: feasible region in which any solution point satisfies all constrains and infeasible region in which any solution point violates at least one of the constraints. The amount

of constraints violation or infeasibility for each solution point could be different. This difference in violation amount is one of the bases for some propose constraint handling methods. The following sections introduce the main constraint handling approaches.

4.3.5.1 Penalized Objective Functions Approaches

The penalizing approach is popular for its success and simplicity. The idea in this method is that the solution points which are infeasible must be penalize so that it appears to be an inferior point comparing to feasible solution points. In this method the value of objective functions corresponding to infeasible solutions are modified so as to make that point an inferior solution point by changing the trend of the primary objective functions. In fact in this method the problem is changed to an unconstraint problem when the constraints are active and some of them are violated.

$$\text{Minimize } f_p(x) \tag{4.6}$$

$$f_p(x) = [f_{p,1}, f_{p,2}, \ldots, f_{p,nf}]^T \qquad x = [x_1, x_2, \ldots, x_{nx}]^T$$

$$f_p(x) = f(x), \qquad \text{if } x \text{ is feasible}$$

$$f_p(x) = f(x) + \text{penalty}, \qquad \text{if } x \text{ is infeasible}$$

$f_p(x)$: *Penalized objective functions*

Several researchers suggested different penalizing strategies depending on the structure of the optimization problems (Dasgupta and Michalewicz, 1997b, Coello Coello, 2002). Three main different strategies for constructing the penalty values are as following:

- Penalize infeasible solution points independent of the amount of violation or any information from the search space by a pre-specified penalizing factor
- Penalize infeasible solutions based on the degree of their violation form the constraints

- Penalize infeasible solutions based on the cost of repairing infeasible solutions such as forcing them back on the feasible region

There is not a generic rules for penalizing infeasible solutions however, some researchers suggested heuristic methodologies such as guidelines formulated in (Richardson et al., 1989) and minimal penalty rule in (Le Riche et al., 1995). However, none of the proposed strategies are universal and difficulties are encountered during handling different optimization problems with specific functionalities and behavior. One could classify the penalizing strategies to three main categories as following:

1. Death penalty (hard constraint handling approach): in this method simply any infeasible solution point generated by the optimization algorithm is rejected and force the algorithm to generate another solution in the feasible region. This method is very simple to implement and is successful in handling simple problems which the feasible region which contain the global optima is convex and covers reasonable portion of the search space. Nevertheless, it fails very often when the problem is complex and the feasible region is nonconvex and the feasible region has a small ratio to the entire search space (this happens quiet frequently in chemical engineering problems) (Dasgupta and Michalewicz, 1997b, Venkatraman, 2004).
2. Static penalty: the strategy of penalizing infeasible solution in this method is to add a penalty factor to the primary objective functions. The amount of the penalty factor is calculated based on the degree of violation (violation amount) from constraints. Although the value of penalty factor is varying depending on the violation form constraint for each solution point however, the penalizing strategy remains unchanged.

$$f_{p,i}(x) = f_i(x) + \sum r_i \Omega(x) \tag{4.7}$$

$$f_{p,i}(x) = f_i(x) + \sum r_{i,j}\, cv_j(x) \tag{4.8}$$

$r_{i,j}$: Penalty parameter of the j^{th} constraint for the i^{th} objective function

r_i: Penalty parameter for the i^{th} objective function

Ω: Overall measure of feasibility

cv_j: Violation of the j constraint

Researchers have been using several methods for measuring constraint violations for formulating penalized objective functions (Homaifa et al., 1994; Michalewicz, 1995b). Penalizing approach for constraint handling has been suffering from a main difficulty regarding assigning an apposite penalizing factor. Assigning too large a penalizing factor can limit the exploration of infeasible region which could result in finding promising feasible solutions while too small a factor could result on wasting computational efforts on evolving an infeasible solution and disregarding feasible solutions. Also, penalizing method changes the trend of objective functions which may cause omitting some nonconvex feasible region. Using penalizing method requires knowledge about the functionality of objectives and assigning penalizing factors are problem specific.

3. Adaptive penalizing strategies: in this class of constraint handling method, the penalty factor is dynamically modified based on some information and feedbacks from the previous iterations of optimization algorithms. This information could be the progress of the algorithm indicated by the number of iteration or some measure of closeness to the promising regions or feasible search area. Some dynamic constraint handling methods have been proposed by (Joines and Houck, 1994; Kazarlis and Petridis, 1998; Hadj-Alouane and Bean, 1997; Smith and Tate, 1993). However, evaluation of appropriate indicators which represents progress of the algorithm or closeness to feasible regions is itself not a simple task. Besides, acquiring these measurements every iteration, sometimes for each point of solutions, reduces the efficiency of optimization algorithms dramatically.

4.3.5.2 *Augmented Objective Functions*

Recent progress in multi-objective optimization methods facilitates application of new approaches for handling constraint which is a challenging task in real world optimization strategies. Infeasibility or violation from constraints can be regarded as a new objective to be minimized simultaneously as the optimization algorithm proceeds. Therefore, extra subsidiary objectives which represent the violation amount form constraints are created to be minimized with primary objectives of the problem. However, application of this method requires an efficient multi-objective optimization method.

$$\textit{Minimize } f_a(x) \tag{4.9}$$

$f_a = [f_1, f_2, \ldots, f_{nf}, \_]^T$ *or* $f_a = [f_1, f_2, \ldots, f_{nf}, c_{v1}, c_{v2}, \ldots cv_{ng+nh}]^T$

$x = [x_1, x_2, \ldots, x_n]T$

f_a*:* Augmented objective functions

cv_j*:* Violation of the constraint j

Ω*:* Overall measure of infeasibility

Variations of this strategy for handling constraints have been studied by several researchers. Surry and Radcliffe (1997) developed a method in which a fraction of solutions are selected based on their ranked objective functions while constraint violation is the basis for selection of the other part of solutions. Deb et al. (2001) compared a technique proposed by Ray and coworker (2001) with four other techniques. In Ray and coworker technique each constraints is treated separately and selection is achieved using heuristic rules with three different non-dominated ranks; constraint violation rank, objective rank, and a rank for combined objective functions and constraint violations. However, implementation of pure form of this strategy has faced difficulties in handling even test optimization problems and there is a lack of evidence for successful performance of it.

4.3.5.3 Approaches Using Heuristics

Heuristic techniques proposed so far are based on the assumption that feasible regions are superior to infeasible region of the search space. In the method proposed by Powell and Skolnick (1993), the heuristic rules developed by Richardson et al. (1989) were applied to map feasible and infeasible region of the search space distinctly so that higher fitness values are assigned to feasible solution points. Deb (2000) proposed a heuristic constraint handling method based on binary relations between feasible and infeasible solutions. In this method the concept of dominancy of solutions are modified to constantly prefer feasible solutions over infeasible solution regardless of the value primary objective functions. As Coello Coello (2002) mentioned this method fails in cases that the ratio of feasible region to the entire search space is small. Pure application of this method is not suggested however, combination of this method with other methods could be helpful.

4.3.5.4 Other Approaches

Challenges in handling constraints in complex optimization problems have obliged researchers to develop many different strategies to handle difficulties in this field. There exit several constraint handling methods which cannot be classified under those mentioned above such as hybrid methods which are a combination of other method, naturally inspired method such as co-evolution and immune system and those originated from classical numerical optimization methods such as Lagrangian multipliers.

A novel hybrid constraint handling method proposed in this work is presented in the next chapter which is a combination of augmented approaches and a heuristic approach which prefers feasible solutions over infeasible solutions.

4.4 Rationale for Development of a New Algorithm

As elaborated in previous chapters the goal of this work is to develop a generic systematic framework for design and synthesis of chemical processes to find the most sustainable design of a process among all possible alternatives. Achievement of this goal requires incorporation of an efficient multi-objective optimization tool which is able to handle a multitude of objectives in a design problem. Computational efficiency of the algorithm is very important in this work due to presence of huge set of nonlinear equation which needs to be solved for evaluation of process alternatives. In addition, as explained before process design problems are highly constrained which necessitates use of very effective constraint handling method. On the other hand a generic framework needs to evaluate processes rigorously to ensure selection of real optimums of a process design problem. This justifies use of a rigorous process simulator to evaluate and design process interactively with the optimization algorithm. Thus, the algorithm must be able to handle black box optimization problems due to interaction with a black box process simulator. The existing optimization problems are not well suited for this task.

Deterministic methods which are considered the most efficient class of optimization methods become significantly inefficient when it comes to handle multiple of objectives. As explained in earlier sections of this chapter these methods were designed for single objective problems and no proper generating strategy exist for deterministic method to generate a diverse Pareto solution. Also deterministic methods are prone to get trapped in local optima although this is not solely for deterministic method and other optimization classes also suffer from this difficulty. However, more importantly deterministic methods cannot handle black box problems without aid of surrogate methods which reduces the reliability of process design which is strongly undesirable.

On the other hand, generating methods such as stochastic and population based optimization method generally suffer from problems such as high computational time and lots of iteration to converge to an acceptable solution, lack of a generic reliable constraint handling method and incapability of handling MINLP problems. Researchers have been proposing different modified

stochastic algorithms to solve the mentioned issues. However, proposed methods do not solve the combination of issues together and each only address some of the mention issues which are essential for achieving the goal of this work. Some of the efforts in this field are as following:

Genetic algorithms (GA), the first population based evolutionary algorithms introduced in literatures, have been applied to a wide range of engineering applications and have showed reliable performance in both discrete and continuous optimization. Some of the most recent studies are: application of GAs for the solution of an MINLP problem by Costaand Oliveira(2001); batch plant design using enhanced multi-objective genetic algorithm by Aguilar-Lasserre et al. (2009); process structure optimization by Yuanet al. (2009); and Genetic algorithms for synthesis of heat exchanger networks by Luo et al. (2009).

Babu and Angira have tested an improved hybrid Differential Evolutionary (DE) algorithm on MINLP problems and concluded it is more reliable and faster than Simplex, Simulated Annealing and GA (Angira and Babu, 2006). Niche algorithms and similar concepts have been used to improve GA. These techniques use a function of sharing distance and density in the population to maintain diverse solutions and approach the Pareto front evenly (Jingjunet al., 2009; Lili D. et al., 2010; Shu, X., 2009). Although this algorithm is very useful for locating solutions in a high dimensional objective optimization, the application of it in chemical engineering is limited and deserves more attention. Although different versions of GA have been proved capable of handling MOO problems with combinatorial search space, these family of algorithms are very slow in convergence and also their efficiency drops in finding good quality solutions when a multitude of objectives are present. This problem is attributed to the algorithm operators which lose their functionality in presence of high dimensional objective space (Deb et al., 2002).

Several algorithms other than GA have been developed over the past 15 years to reduce the processing time of population based algorithms. Four of the most popular ones are memetic algorithms (MAs), particle swarm optimization (PSO), ant colony system (ACS) and shuffled

frog leaping (SFL). Elbeltagi et al.(2005) have discussed performance of these algorithms. They showed that PSO is the most powerful in terms of success rate and solution quality. Yiqinget al., (2007) applied an improved PSO algorithm for solving non-convex process synthesis problems (Yiqing et al. 2007). A strategy based on dynamic neighborhood concept was developed by Xiaohui and Eberhar (2002) for handling multi-objective problems (Xiaohui and Eberhar 2002). This method needs a priori knowledge of objectives. Parsopoulos and Vrahatis (2002) adopted the idea of vector evaluated GA in PSO to handle multi-objective problem. Bartz-Beielstein et al. (2003) enhanced PSO for Pareto optimization using archiving technique. Ho and Shiyou et al. (2005) have defined a new fitness value for each particle based on the number of particles it dominates to adopt PSO for multi-objective problems.Also concepts such as crowding distance are used in multi-objective PSO algorithms to improve its performance (Raquel and Naval, 2005). Based on the previous studies and comparisons PSO has shown better performance in terms quality of solutions and convergence rate however, original PSO is neither capable of handling multi-objective problems nor MINLP problems.

Next chapter is allocated on developing a novel PSO based algorithm which assures the following features necessary for the interest of this work.

- Capable of handling a multitude of objectives without losing its efficiency.
- Converging to a good quality solution set in a reasonable time which enables handling large scale process design problems
- Handling black box optimization problems which enables utilizing sequential process simulators to ensure reliable design and reduce equality constraints associated with process design problems
- Capable of handling MINLP optimization problems
- Incorporating a generic and reliable constraint handling method

General rationales on choosing PSO algorithm as the basis for developing an appropriate algorithm in this work is listed below.

- PSO algorithm like other population base algorithms evolves a collection of solutions at a same time instead of jumping from one location of the search space to another location which enables it to attain more than one solution at every iteration to approximate Pareto solution.
- It is more efficient than other population algorithm in terms of computational burden and time.
- Computational parallelization which could reduce computational time significantly can be applied effectively with this method due to the existence of a swarm of suctions.
- A swarm of solutions evolving in the search space decrease the likelihood of getting stuck in local optima.
- It is a methaheuristic method independent of regular auxiliary information such as derivatives, Hessian matrix for indirect optimization algorithms. This enables the algorithm to handle black box problems conveniently.
- Independency of gradient of objective functions and constraints makes the algorithm more effective in handling mixed integer and combinatorial search spaces and also nonconvex nonlinear multimodal optimization problems can be handled conveniently with this algorithm comparing to deterministic methods.

CHAPTER 5

MULTI-LEADER MULTI-OBJECTIVE PARTICLE SWARM OPTIMIZATION (MLMOPSO)

5.1 Introduction and Background

Decision making about multi-objective problems is always involved when designing or modifying a chemical process. A common example is to maximize profit while minimize waste in a chemical process. These two objectives contradict each other and result in a multi-objective problem. Increasing focus on environmental, social and economic concerns imposes more objectives on design problems. Traditionally, all objectives of a problem would be combined into a single objective problem based on the preferences of a decision maker. However, using this approach for solving multi-objective problems depends on the knowledge and preferences of the decision maker. It becomes a challenge to handle problems with high dimensionality of objectives with essentially different characteristic and magnitude using the single objective approach.

Optimization algorithms are roughly divided to either deterministic or heuristic. Among heuristic methods, population based metaheuristic methods seem to be more effective and capable of handling complex problems. Deterministic methods are usually based on gradients of the objective functions and constraints while metaheuristic methods search the feasible solution area using a random search algorithm. Both approaches have been used for large-scale industrial process optimization.

Whereas deterministic methods can be used for preference based methods, metaheuristic methods are better suited for generating methods (Grossmann and Guillen-Gosalbez , 2009). Many deterministic methods have been used to address environmental criteria in processes by using a single objective approach. However, their application to multi-objective optimization is limited so far. The other engineering disciplines such as electrical and mechanical engineering have applied

Metaheuristic population based methods simultaneously handle a set of trial solutions and generates a set of non-dominated solutions in a single iteration while other methods must perform a series of iterations. Population based algorithms are also capable of handling complex functionality such as discontinuities, multimodality, disjoint feasible space and noisy function evaluations. They are less dependent on the shape of the objective function and constraints which is a big problem for deterministic algorithms. In addition, conventional techniques such as gradient based, simplex based algorithms and simulated annealing were not designed for multi-objective optimization (Fonseca and Fleming, 1995).

Genetic algorithm (GA), one of the first evolutionary algorithms (such as GA, evolution strategies or Genetic Programming), and variations of it have been widely applied in chemical processing problems. Some of the most recent studies are: application of GAs for the solution of a multi-objective optimization problem (MOO) problem (Costa and Oliveira, 2001); batch plant design using enhanced multi-objective genetic algorithm by Aguilar-Lasserre et al. (2009); ordinal Ranking Genetic Algorithm (ORGA) by Jin X. (2005) optimizing the William Otto process. Also Angira and Babu (2006) and Babu et al. (2005) have tested an improved hybrid Differential Evolutionary (DE) algorithm on several chemical optimization problems.

Optimization problems with high dimensionality in objective functions are very common in chemical processing especially when environmental and social issues have to be accounted for in

the design. Evolutionary algorithms use operators for selecting the non-dominated solutions to maintain good distribution in the solution set. As stated by Deb et al. (2001) when the dimension of objective space is increased the efficiency of operators which are responsible for maintaining good distribution in the solution set decreases. For example the operators which are based on crowding distance in a neighborhood such as niching operator encounter problems due to dimensionality. In the higher dimensional space (more than two) the definition of the neighborhood of a solution along the Pareto curve is not clear and even in the case of having a clear definition for neighborhood of a solution, calculating distance metrics from all neighbors for each solution becomes computationally expensive and reduces the efficiency of the algorithm (Deb et al., 2001).

Particle Swarm Optimization (PSO) is one of most efficient methods in terms of success rate and solution quality and has been the focus of researchers in past decade (Sierra and Coello Coello, 2006). The concept of PSO is inspired from social behavior of colonies such as a flock of birds or a school of fish. In the regular single objective PSO a population of particles move toward a global best (leader) using their past experience from the past iterations and search the decision space. Their speed toward the leader is varying randomly to ensure searching for the global optimum.

Unlike evolutionary algorithm, original PSO which is not an operator based algorithm although there are some parameters associated with it may be adopted for multi-objective optimization to solve the problem of high dimensionality and high nonlinearity of design objectives in chemical engineering. Recently, researchers have been using PSO to handle multi-objective problems due to its good performance. PSO is faster and more efficient in terms of solution quality and success rate (Elbeltagi et al., 2005). The main issue with using MOPSO (multi-objective Particle Swarm Optimization) is selecting a guide in the swarm (leader) for the particles in the swarm. Handling multi-objective optimization problems with PSO needs

redefinition of a leader in order to find optimal non-dominated solution set. However, selecting the leader from non-dominated solution set is a profound task to maintain diversity and attain convergence. To address this issue, researchers have adopted different ideas such as sigma method (Mostaghim and Teich, 2003), dynamic neighborhood (Hu and Eberhart, 2002), elite particles in hypercube (Coello Coello et al., 2004) and archiving method (Ho et al., 2005).

However, the application of PSO especially multi-objective PSO in the field of chemical engineering has been limited comparing to the other engineering disciplines. A new method for updating the position of particles in the swarm is proposed in this work in which there is no need for calculating any operator and the particles select the most similar non-dominated solutions as their leaders based on the new form of the updating equation. In addition the new form of the equation incorporates the information of several leaders thus enhances the search the space between leaders which in turn results in maintaining more diversity and good distribution in the final solution set.

In this work, the main objective is to introduce a novel method for updating particle positions in the swarm for handling multi-objective problems and to address the aforementioned issues such as maintaining diversity in the final Pareto set. The proposed method is compared with the main existing techniques for adopting PSO to multi-objective problems. This method which uses the information of more than one leader to update position of particles is called MLMOPSO (multi-leader MOPSO). The implementations of this algorithm on three different test functions, well explored by other MOPSO techniques, show that the proposed algorithm (MLMOPSO) can converge to a very high quality approximation of Pareto front with a very good diversity. Also this algorithm was compared with two other MOPSO algorithms that have the same structure as MLMOPSO but with different method of selecting and applying leaders (social information). A modified archiving technique which is used in MLMOPSO was used for the other two methods (Sigma method and oneleader method) in order to have a justified comparison between them.

5.2 Definition of Multi-Objective Optimization

A minimization multi-objective problem is defined as:

$$\text{Minimize } \{f_1(x), f_2(x), \ldots..,f_n(x)\} \quad n> 2, \tag{5.1}$$

$$\text{Subject to } x \in A$$

$f_i : R^m \rightarrow R^n$ that are to be minimized simultaneously are conflicting objectives. Two objectives are conflicting when improving one of them cannot be done without deteriorating the other one. A decision vector, $x = (x_1, x_2, \ldots,x_m,)$, is the vector of decision variables which are belong to feasible region, $A \in R^m$. The image of the feasible region of decision space on the objective function space, $Z \in R^n$, is called feasible objective region. The elements of feasible objective region are objective vectors consisting objective components, $f(x) = \{f_1(x), f_2(x), \ldots..,f_n(x)\}$.A member of decision space $x_1 \in A$ dominates $x_2 \in A$ (showed $x_1 < x_2$) iff:

- All the component of decision vector $f_i^1(x)$ is less or equal to their corresponding component in $f_i^2(x)$; $f_i^1(x) \leq f_i^2(x)$, $i=1,\ldots., n$. this also is known as weakly dominancy of x_1 over x_2.
- At least one of the component in $f^1(x)$ is less than its corresponding component $f^2(x)$; $f_i^1(x) < f_i^2(x)$ for at least one $i=1,\ldots., n$.

A decision vector is called non-dominated in a sub region, Z, of the feasible decision region, A, if there is no another decision vector in Z which dominates it. A set of decision vectors which all its members are non-dominated in the entire feasible decision space A is called Pareto Optimal set. The image of Pareto Optimal set on the objective space is called the Pareto Optimal front.

5.3 Particle Swarm Optimization

The Particle Swarm Optimization method is a population based optimization technique. A group of N particles, called "The Swarm", achieve improvement at each iteration based on their background and experience. Each particle has a position which represents a decision vector in the decision variable space $x^i = (x^i_1, x^i_2, \ldots, x^i_m,)$ and a vector of velocity defined by $v^i = (v^i_1, v^i_2, \ldots, v^i_m,)$. In each iteration each particle updates its current position using the information of the best position it ever had, called "The Personal Best", and the information of the best position have ever been found in during past iteration by all particles in the swarm, called "The Global Best." The equations used for updating position of particles are:

$$v^i_{j,,t+1} = w\, v^i_{j,t} + c_1 r_1 (G^i_{j,t} - x^i_{j,,t}) + c_2 r_2 (P^i_{j,t} - x^i_{j,,t}) \tag{5.2}$$

$$x^i_{j,t+1} = x^i_{j,,t} + v^i_{j,,t} \tag{5.3}$$

In Equation 5.2 and 5.3, t, is the iteration number, P and G respectively represent Personal Best and Global Best, w is the inertia constant, c_1 and c_2 are two positive constants and, r_1 and r_2 are two random values in between [0,1].

Indeed, P in Equation 5.2 represents the memory of each particle of the best position it has found ever and G represents the best experience of all particles in the swarm. The comparison of the performance of particles is based on an evaluation function which is pre-defined and related to the objective functions of the optimization problem. The global and local search ability of the algorithm is controlled using inertia constant which is the influence of the previous velocity on the current velocity of each particle.c_1 and$c_2$control the impact of social and personal memory and r_1 and r_2 ensure the random movement of the particles and global search ability of the algorithm (Mostaghim and Teich, 2003; Shi and Eberhart, 1998).

5.4 Adopting PSO for Multi-Objective Problems

In a single objective optimization problem the global best is determined easily by finding the best position in the swarm. However, for a multi-objective problem because there is a set of non-dominated solution in the swarm, at least one non dominated solution (called *leader* here) must be selected to guide each particle toward the Pareto front. This is the major task in Multi-Objective PSO (MOPSO) and has a large impact on the diversity and quality of the Pareto Solution.

In the past decade researchers have proposed several methods for adapting PSO for multi-objective problems (Bartz-Beielstein et al., 2003; CoelloCoello et al., 2004; CoelloCoello and Lechuga, 2002; Mostaghim and Teich, 2003). All the successful approaches have a common feature that they use an external repository, called *archive*, for keeping the best non-dominated solutions obtained during past iterations. A maximum number is set to avoid the *archive* getting too large. The leaders for updating the position of particles in the swarm are selected from this *archive*. The major difference between these methods is how to select the leaders to maintain diversity and obtain a good approximation of true Pareto front. The basic steps of MOPSO algorithms are presented in Figure 5.1.

At each iteration, denoted by *t*, particles are evaluated by objectives functions and non-dominated particles are compared with the members of the *archive* and dominated members are removed in order to maintain only non-dominated solutions in the *archive*. A non-dominated set is a set where no two pair of its members dominates each other. To avoid the *archive* size becoming too large, a fitness function is used to assign a fitness value to each member of the *archive* and the members with higher fitness are selected to remain in the *archive*. The functionality of this fitness function is to keep the most successful members which preserve more diversity in the *archive*. In the next step, based on the MOPSO strategy either one leader is selected to update the position of all particles or a different leader is selected to guide each

particle. The common strategy for selection of the leaders will be explained briefly in the next section of this chapter. According to Equation 5.2 and 5.3, to update the position of each particle, a Personal Best of each particle is needed as well as a leader. A Personal Best for each particle is compared with its current position at each iteration and the non-dominated one is selected as the new Personal Best. In this work, a novel technique for updating position of the particles is introduced and modified approaches are utilized for updating the *archive*, selection the leaders and Personal Bests. Also a modified stopping criterion is used to terminate the algorithm.

1- t=0;
2- Initialize position of particles in the swarm and assign an initial velocity for each x_t^i, v_t^i ..
3- Initialize the *archive* as: A= [].
4- Evaluate particles in the swarm based on objective functions and update the *archive* with the new non-dominant ones.
5- Select leader or leaders from the *archive* using a selection method.
6- Update velocity and position of the particles using Equation 5.2 & 5.3.

Figure 5.1- basic steps of Multi-Objective PSO algorithm

As mentioned in the previous section several methods exist for handling multi-objective problems with PSO algorithm. Although they have some differences in the main algorithm all of them use the same structure presented in Figure 5.1. The method for the selection of leaders in the swarm is the main difference between algorithms. In this section, some of these algorithms are briefly explained, and then a new approach is introduced.The main objective of different selection methods is to maintain diversity and quality of the final Pareto set. A new PSO based optimization method, called MLMOPSO, is proposed in next sections which uses a new equation for updating particles position using several leaders. Also a modified archiving technique is

utilized in this work to ensure capability of the algorithm in handling many objectives, preserving diversity in the solution set and the convergence quality of final solution set.

In work by Hu and Eberhart (2002) an algorithm called the dynamic neighborhood was proposed. In this algorithm only one objective is optimized at a time while the values of the other objectives are kept fixed. One-dimensional optimization is used to optimize the unfixed objective. The local optimum among k neighbors in terms of the unfixed objective is used as the global best. This approach is useful when there are a few objectives to be optimized. Also a priori knowledge about the problems objective is required to select fixed objectives. In their work this algorithm was explained for two objectives.

Coello Coello et al. (2004) presented a new approach that incorporates the concept of Pareto dominancy in the swarm. This algorithm uses a secondary repository of particles that is used with other particles in the swarm to update their positions. They also incorporated a mutation operator inspired from evolutionary algorithms to extend the exploratory capability of the algorithm.

Bartz-Beielste et al. (2003) used an *archive* base algorithm for handling multi-objective problems using PSO, called DOPS. They adopted well-known archiving technique form evolutionary algorithms into PSO to maintain diversity among the non-dominated set. Deletion fitness and selection fitness are assigned to each member of the *archive*. Each time a global best is needed in the swarm a particle from the *archive* is selected based on its selection fitness. If the number of non-dominated members surpasses the *archive* size deletion fitness is used to remove some member from the *archive*. The selection and deletion fitness are defined based on the influence of each particle on the diversity and each particle success rate of being non-dominated.

Mostaghim and Teich (2003) proposed an MOPSO which incorporates a sigma value to select a leader for each particle in the swarm. A sigma value is assigned to each particle in the swarm and the closest non-dominated solution in terms of sigma value is selected as the leader for

each particle. This method may result in clustering around some point. The author used a turbulency operator to increase the exploratory capability and decrease the probability of clustering.

Other researchers also employed concepts such as clustering, multiple sub swarms with migration operators as well as adopting evolutionary based operators (Leong and Yen, 2008; Goh et al., 2010). However, utilizing these concepts in MOPSO algorithms gives more complexity and reduces the efficiency of the algorithm to be applied in a real practical optimization problem. Detecting clusters in the swarm or incorporating evolutionary operators need calculation of extra indicators and operators at each iteration which impose more computational expense on the algorithm. This becomes very important while dealing with large problems. The reader is referred to the work by Pulido and Coello Coello (2004) for more information on clustering approaches and using sub population. As it can be seen Pulido and Coello Coello (2004) use an inner loop for each sub swarm which imposes more computational expense on the algorithm. In addition in their work performing clustering step after each inner iteration in order to select swarm leaders further imposes computational expense on the algorithm. In their work, the leaders are selected using clusters detected based on distance between particles and similarity in the decision space. Similarities and distance between particle and leaders are considered in the new form of updating equation proposed in this study without performing clustering. The work by Jin X. (2005) can be consulted for more information on calculating evolutionary operators such as niching operator and its computational expense in multi-dimensional spaces.

The algorithms such as the sigma method presented by Mostaghim and Teich (2003), the algorithm presented by Fieldsend and Singh (2002) or method proposed by Coello Coello and Lechuga (2002) select a particular leader for each particle based on a criterion of closeness which is a representative of their analogy. However, in such a way of selecting the leaders, some area in between of the currently selected leaders may remain unexplored and some type of perturbation

and turbulacy operator are needed to throw particles in unexplored region. This imposes more computational burden to the algorithm. On the other hand, some algorithms use only one leader for all particles in the swarm at each iteration such as the algorithm proposed by Bartz-Beielstein et al. (2003). They select the leader based on roulette wheel or random selection from the *archive*. Changing the leader of the whole swarm at each iteration may cause perplexity in the swarm at the early iterations at least for some particles and clustering in the final solution set. And it prevents exploring the search area toward one leader sufficiently. Also there is more probability of not seeing search area around the *archive* members which haven't been selected although they would have good potential of leading particles toward solutions with high quality.

5.5 New Approach of MLMOPSO

5.5.1 Main Features of MLMOPSO

A novel approach is proposed in this work to ensure diversity in the final Pareto solution set and its convergence to the true Pareto set especially while dealing with high number of objective. The main differences in the method being proposed can be presented as following:

- A new equation is utilized for updating the position of particles in the swarm at each iteration. According to Equation 5.4 each particle uses the information of several chosen leaders for updating its position. Each particle flies toward the non-dominated set at each iteration using a velocity vector. This way particles use information of several leaders instead of only one closest leader to maintain diversity. The more the algorithm progress toward the Pareto front the more the closest leader influences the velocity vector. The amplitude of this influence is dependent on the value of α which is defined as the *Social Influence Factor* (SIF). The smaller SIF the less influential are the leaders other than the closest leaders to a particular particle to contribute to its velocity vector. If the value of the SIF is too small this algorithm performs the same as other MOPSO algorithms which

use only one leader for each particle. While a particle is far from the non-dominated solutions the information of more leaders contributes to its flight path therefore it flies toward the position which is closer to colony of non-dominated solutions. As the particle reaches sufficiently close to the front of leaders if this front is evenly distributed, the influence of multiple leaders is kept otherwise the influence of the closest leader would be increased. Also according to the new form of the equation of updating velocity, even when a particle reaches to the position of very close to a leader it still can move toward the other leaders. However the SIF factor should be large enough to allow the particle to follow leaders other than the closest one to the particle. When an area in the decision space is such that encompasses several optimum solutions using a larger SIF enables the particles to explore the area around and between the obtained optimum and increases the probability of getting improved toward true Pareto front. Magnitude of the SIF can be adapted as a tuning factor which controls the influence of several leaders rather than only the closest one.

$$v^{i}_{j,,t+1} = w\, v^{i}_{j,t} + c_1 r_1 (P^{i}_{t} - x^{i}_{t}) + c_2 \sum^{n}_{j=1} r_j \beta^{i}_{j} (L^{i}_{t} - x^{i}_{t}) \tag{5.4}$$

$$\beta^{i}_{j} = exp(-(d^{i}_{j}/\delta^{i}_{j})^2)/(\sum^{n}_{j=1} exp(-(d^{i}_{j}/\delta^{i}_{j})^2)) \quad , \quad \delta^{i} = \alpha\, d^{i}_{min} \tag{5.5}$$

In Equation (5), P^{i}_{t} is the Personal best for particle i, L^{j}_{t} is the leader jth in the swarm, n is the number of leaders, d^{i}_{min} is the minimum normalized distance between particle I and leaders, $d^{i}_{j}$is the normalized distance between particle i and leader j and r_i is a random number. Other parameters are defined same as parameters in Equation 5.2.

- In MLMOPSO the distance between particles and leaders is evaluated in the decision space and in means m-dimensional Euclidian distance. Particles move and explore the decision space and their aim is to fly to the position which its image in the objective space is non-dominated. If the normalized distance of a particular particle in the decision

space is closer to a particular leader solution it means that particle is more alike the mentioned leader in terms of decision space properties. Therefore it is more likely for a particle to get to a non-dominated position if its normalized distance in the decision space is closer to leaders meaning the closer a particle is to a leader in the decision space the more its characteristic is alike that leader.

- The form of Equation 5.4 helps exploring the space between the currently selected leaders. At the early iterations each particles flies using a flight vector toward several leaders not only one leader. SIF parameter and the random numbers in the Equation 5.4 allow flying sometimes more toward leaders other than the closest one. This would avoid the problem mentioned about the algorithms selecting the closest *archive* member as the leader for each particle. At the last iterations because the particles are close to the leaders the influence of the closest is more however having a proper SIF factor along with the random coefficients allow the particle searches between leaders and improves the Pareto front even at last iterations. Also a proper SIF would decrease the clustering phenomena due to allowing the particles to fly toward other leaders and locating new non-dominated members in Pareto front if there is any.
- A fixed number of leaders are selected and used for updating the position of particles. In order to select the best solutions ever found during past iterations the leaders, non-dominated particles are kept in an external *archive* and the *archive* members are updated at each iteration. The *archive* size is limited to avoid increasing tremendously in complex problems. The archive members will be selected in accordance to aggregated fitness assigned to each particle in the *archive*. The aggregated fitness is a weighted summation of f_{sel} and f_{del}. f_{del} is a fitness value for deleting each member from archive defined based on diversity in the *archive* so that deletion of a member with the large f_{del} helps to maintain diversity. Also f_{sel} is defined based on success rate to keep successful members in the *archive*. To avoid expelling successful leaders or good Pareto solutions in sparse

regions from the *archive*, the aggregated fitness is used which is a weighted combination of f_{sel} and f_{del}. This accounts simultaneously for success rate and diversity of the members. Use of the aggregated fitness function for selection of the *archive* members results in preserving diversity in the set of non-dominated solutions in the *archive*.

- The *archive* members are sorted based on their aggregated fitness. In order to select the leaders a desired number of leaders will be selected from the top sorted *archive* members. Until the number of *archive* members is not more than desired number of leaders, all the *archive* members are used as the leaders.
- Local best of each particle will be selected based on the concept of non-dominancy. In the case that the current and previous best position of a particle are both non-dominated, a fitness value based on the number of particles dominated by each is used to select one.

5.5.2 Deletion and Selection Fitness Functions

Two different methods were used for evaluating the deletion fitness of each particle. The first method, called adaptive grid was proposed by Knowles and Corne(2000) and its variations used in (Coello Coello et al., 2004) and (Bartz-Beielstein et al., 2003). In this method the objective space is separated to hypercubes which creates the grid. The edge length of each hypercube is calculated using Equation 5.6 (Bartz-Beielstein et al., 2003). The grid size will be reevaluated whenever a new non-dominated particle is fined in the swarm and the number of non-dominated solutions is more than maximum *archive* size. The computational burden of this method is a lot less than niching method (Knowles and Corne, 2000).

$$Cubeedge_i = c(max_i - min_i)/swarmsize \quad (5.6)$$

In Equation 5.6, max_i and min_i are the maximum and minimum of objective function i^{th} reached by the member of the *archive*. And c is a constant, $c \in [0\ 1]$, and the smaller value of c the less is the probability of sharing a certain hypercube with two particles. Therefore, higher

values of c increase the pressure of deletion fitness which is defined as the squared reciprocal of the number of members which share one hypercube, H.

$$f_{del} = H^2 \tag{5.7}$$

The second method used for evaluating f_{del} is based on the relative distance of the *archive* members.

$$f_{del} = \sum^{n}_{i \neq j} 1/D_{ij} \tag{5.8}$$

$$D_{ij} = \left(\sum^{k}_{j} (OF^{i}_{k} - OF^{j}_{k})/(max_{OFk} - \min_{OFk})\right)^{0.5} \tag{5.9}$$

D_{ij} is the distance between *archive* member i and j, OF^{i}_{k} is the value of objective function number k of the i^{th} *archive* member. The problem of this method is high computational burden with order of $O(n^2)$ (Bartz-Beielstein et al. 2003).

Both of the mentioned method, Hypercube and distance based f_{del}, help to maintain diversity in the final solution set and exploring the sparse regions because they pay off more the members located in sparse regions and are far from the others. However, to prevent expelling solutions with good quality and history the selection fitness, f_{del}, also contributes in evaluating aggregated fitness. The number of time that a particle is selected as an *archive* member is recorded and normalized to be used as the selection fitness value to reward *archive* members with high success rate. At each iteration only if the number of non-dominated solutions surpasses the maximum number of the *archive* or there is a new non-dominated solution from the swarm the aggregated fitness is reevaluated. In this algorithm no repetitive non-dominant solution is allowed to enter the *archive*. Therefore there is no repetitive solution in the *archive* to confound the performance of selection and deletion methods.

5.6 Comparison Methodology and Performance Metrics

The proposed MLMOPSO method is validated using three test functions obtained from literature. These test functions are presented in Section 10. The performance of MLMOPSO was compared with two other methods; First, sigma method proposed by Mostaghim and Teich (2003) which at each iteration selects one leader for each particle based on the sigma value, Second, the method proposed by Bartz-Beielstein et al.(2003) which selects one leader for all particles in the swarm, called One-leader, using fitness values that maintains diversity in solution set. Each of these methods is a representative of two main concepts of assigning leaders in the swarm.In order to compare and evaluate the performance of each method four different metrics taken from literatures were used. Although NOFE could be a good representative of algorithm efficiency for single objective optimization, in multi-objective optimization because there is more than one non-dominated solutions the diversity, distribution and quality of the solutions along the true Pareto front is very important. Therefore, it is important to evaluate the quality and diversity of the solution set using other metrics (Deb, 2001). A variety of performance metrics have developed for this purpose which four of them are explained and used in this study.

S-metric: In the S-metric (Zitzler et al. 1999) the hyper volume of the multidimensional objective space enclosed between the solution set and a reference point is calculated. This metric is used to measure the diversity and convergence of the obtained non-dominated solution set. The reference point for comparison of two final solutions must be same otherwise one solution set can have different S-metric depending on the chosen reference point. S-metric is used to compare the diversity and closeness of the solution set. A formulation was given by Bartz-Beielstein et al. (2003) for calculating the S-metric in a minimization problem which is the communion of volumes dominated by each member of the non-dominated solution set and bounded by the reference point.

Weakly dominancy criteria: To compare the performance of each method in converging to the Pareto front a metric that is called weakly dominancy criteria, C(A,B), is used. This metric shows the percentage of solutions in set of A which are not weakly dominated by the solutions of B. it should be considered that necessarily 1- C(A,B) is not equal to C(B,A) and both solutions should be considered (Raquel and Naval 2005).

$$C(A, B) = |\{b \in B; \forall a \in A: a \geq b\}| / |B| \quad (5.10)$$

Spread: this metric may be an equivalent to the S-metric and is used here to show width and spread of the Pareto front obtained by the algorithm. The spread is calculated as the length of diagonal of the largest hypercube enclosing the solution set. A large spread is desired since it shows diverse trade-off solutions. It should be taken to account the presence of few isolated solutions in sparse region can make the spread larger because spread is calculated on based on the extremes. In the case that the true Pareto front is known spread larger than true Pareto front multi-dimensional diagonal is a representative of error and closeness of the solution set to the true Pareto front (Goel and Stander, 2010 b).

Non-Uniformity measure: this measure is to complement spread metric and detect the presence of poorly distributed solutions. This metric estimate how uniform is the solution set. This is called measure non-uniformity metric since it is the standard deviation of the crowding distance of solution set members(Deb, 2001). The smaller is the non-uniformity the better is the quality of the solution set. This measure is calculated as:

$$\Delta = \sum_{i=1}^{N} |d_i - d|/N, \quad d=1/N\sum_{i=1}^{N} d_i \quad (5.11)$$

Where, d_i is the crowding distance of the final solution set in the objective space.

5.7 Parameter Setting

In order to test the proposed algorithm and compare its performance equitably with two other approaches first the other two methods both methods were modified. Sigma method was modified in terms of *archive* member selection in the case of having more members than the maximum size of the *archive*. For the selection of the *archive* member the same strategy as MLMOPSO method using aggregated fitness function was used for sigma method. The archive member selection method was based on the strategy presented in (Bartz-Beielstein et al., 2003). The population was initialized for all three methods using Hammersley sampling method which is explained in section 9. The maximum allowable archive size for all method was 200 and the swarm size was 100. Inertia constant, w, was set to 0.8 and c1 = 2.8, c2 = 1.2. The reader is referred to the work by (Bhattacharya and Samanta, (2010); Shi and Eberhart, (1998) for further information on selecting PSO parameters. The algorithms would be terminated if a novel stopping criteria was met before reaching to a maximum number of iterations.

For this work the number of leaders contributing to particles updating equation was limited to 30. For the number of leaders between 2 to 30 a SIF value between [0.2 8] is recommended. As it is mentioned earlier normalized distance is used in the updating equation to account for all dimensions justly. The ratio of d^i_j / d^i_{min} represents the relative distance of other leaders to a particular particle over the closest leader distance. The radius in which the leaders may contribute in leading a particle is defined by the range of this relative distance. However, the contribution of each leader in the mentioned ranged is determined by the value of SIF factor, α. The ratio of $d^i_j / (\alpha\ d^i_{min})$ is called intensified relative distance. SIF, α, intensifies the range and the actual radius of contributing leaders is inversely related to SIF. Thus, if SIF is small it means that only the leader which has a very small relative distance (almost in a same distance of the closest leader and similar to the closest leader) contribute to the updating equation.

Vectors of random relative distance between (1 15] were generated 100 times. This procedure was repeated for the number of leaders in the range of [2 30]. It was observed that the lower limit for SIF of 0.2 for all relative distance in the mentioned rage results in only contribution of the leaders almost in a same distance of the closest leader. On the other hand the upper limit of 8 is recommended for moderate number of leaders as this value almost evenly distributes the contribution between leaders for the range of relative distance between (1 15]. However, larger numbers can be assigned for SIF which allow contribution of farther leaders. It is recommended to adapt the value of SIF based on the progress of the algorithm and increase SIF dynamically as the algorithm finds solutions closer to the true Pareto optimal front. However, Dynamic adaption of SIF and detail exploration on the effects of different ranges of SIF is out of scope of this study and is subject of the other work by the Authors. In this work a fixed SIF was used. It is recommended to define SIF in the range of [0.2 8].

SIF factor for MLMOPSO was set to 3. This value makes the algorithm use the information from the leaders other than the closest leader moderately. No special operator was applied in any of algorithm to analyze and compare the effect of using new updating strategy and aggregated fitness values in maintaining the diversity in the solution set. In this work, weighted coefficient of f_{sel} and f_{del} were respectively 0.3 and 0.7 for calculation of aggregated fitness function. Also the adaptive grid method was preferred for evaluation of f_{del} because of its higher efficiency. Although a new stopping criteria was used to terminated the algorithm an upper limit for the number of iterations was 100.

Having a good stopping criterion helps to compare the algorithm based on the minimum number of iteration that they can obtain a good quality solution. The stopping criterion was activated for some implementations in this work to compare the algorithms in terms of number of function evaluations (NOFE).

5.8 Stopping Criteria

New stopping criterion was applied on the *archive* which holds the best non-dominated solutions ever found during the iterations. A quantitative measure was developed for dynamically checking the steady state condition of the *archive*. The basic idea of the stopping criteria is to have a solution set that is not changing any more with more iteration. A not changing non-dominated solution should keep a certain shape that has a Centroid movement less than a predefined margin. Variances of the objective function values of the *archive* members are calculated. Change in the average of objective variances is accounted as a complementary indicator to be less than a predefined value to ensure having a not changing solution set. The objectives were scaled in order to calculate the variance and the Centroid of the *archive*. Both mentioned criteria, called statistical criteria, must be met for the *archive* for a predefined consecutive number of iterations which is 10 in these implementations to terminate the algorithm.

In order to ensure convergence to a mature final solution in addition to statistical criteria for termination of the algorithm a non-dominated based criterion introduces by 1 and Stander (2010 a) must be met. In their approach the *archive* of current iteration is compared to the *archive* of y^{th} previous iteration (y is the iteration gap) and Consolidation ratio (*CR*), the ratio of the solutions in the *archive* of y^{th} previous iteration which also present in the current *archive*, is calculated. If CR is more than a threshold and also statistical criteria are met the algorithm is terminated. The iteration gap acts as a filter for noise if this value is very small it may result in a pre-mature solution. In this implementation y was 10 as it is recommended in (Goel and Stander, 2010 a) and the threshold for *CR* was 0.85.

Once the *archive* meets all three criteria the algorithm is terminated. This helps to avoid the problem of oscillation in the Pareto set and not incorporating a priori knowledge about the final solution or a fixed number of iteration. In some situations, a fixed iteration number would result

in excessive iterations or stopping the algorithm prematurely. Implementation of the mentioned modifications on optimization algorithms improves the efficiency and reduces the processing time.

5.9 Initialization Method

The complexity of optimization problems is due mostly to dealing with highly nonlinear objective functions with multi-modal behavior of the surface. The other complexity of these problems is due to size and dimension of the problem domain. Initialization of population based algorithms has a very significant effect on the quality of the final solution. Generally initial population is initialized using random sampling methods that often results in over sampling of some region and sparse sampling of some other regions.

A low discrepancy method, called Hammersley, is used to sample the search space and generated the initial population. It samples the search space more uniformly than the other random based method. The Hammersley sampling method is the optimum way of placing n points in a k-dimensional space. There are other techniques to uniformly distribute points in a k-dimensional space such as Latin-Hypercube method. This method has been design to locate points uniformly along a line and then randomly pair up for k-dimensional space (Diwekar and Kalagnanam 1997). The uniformity property of this sampling method is compared to random sampling and hypercube sampling method for a two dimensional search surface in Figure 5.2.

a) Hammersley sampling b) Latin Hypercube c) Random sampling

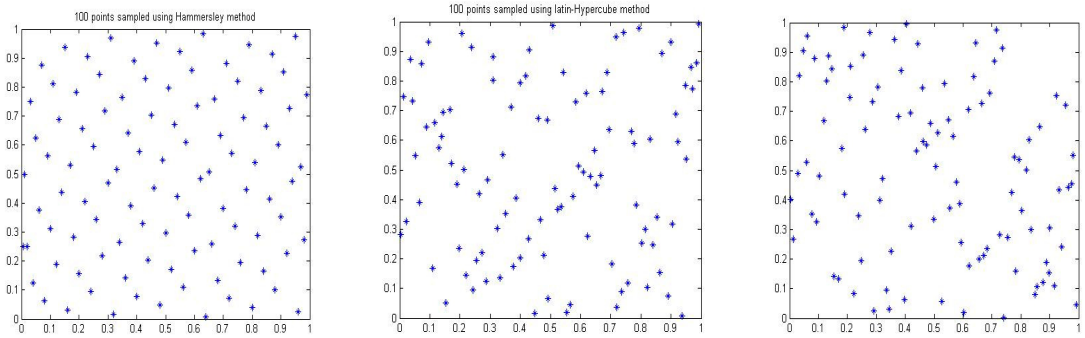


Figure 5.2- uniformity property of sampling a 1 by 1 decision space with 100 points using three sampling methods; a) Hammersley b) Latin-hypercube 3) Random

Figure 5.2 shows the Hammersley method samples which are a lot more uniform than the other two methods. Therefore the preferred method used in this work for initialization of the population for all three MOPSO methods was Hammersley method.

Table 5.1-Summary of the features incorporated in MLMOPSO algorithm

An external *archive*	To preserve all good quality non-dominated solutions obtained by the algorithm. To provide the possibility of selecting best solutions as the leaders in the swarm.
Fitness functions	To screen and rank the solutions based on the diversity and success rate indicators. To select the high rank solutions in the *archive* as leaders in the swarm to enable more diversity and wider search capability.
Information of several leaders in the swarm is used to guide each particle	To maintain more diversity and avoid clustering in some limited region.To explore more the search space even between non-dominated solutions by changing the vector of velocity toward different leaders. To increase exploring of the feasible regions around non-dominated solutions as algorithm proceeds toward the end.
Use of a novel equation for updating the position	To minimize the computation burden for incorporating the information of the leaders in the vector of velocity of each particle.
Calculation of the distance of the leaders and particles in the decision variable space	To increase the efficiency of the algorithm by making the particle follow the leaders with more similar characteristics.
Selection of local best	To select a local best for each particle with the best quality in the swarm

using dominancy concept	among possible previous positions.
Use of an *archive* based stopping criteria	To detect any improvement in the solution set using two different type of indicators. To avoid the computational expense imposed by using a pre-specified maximum iteration number.
Use of Hammersley sampling method instead of random initialization	To sample the whole search space evenly in order to avoid having a part of space without any initial particle. To increase the efficiency and consistency of the algorithm.

5.10 Test Functions

Three test functions selected from (Zitzler et al.,1999) have been used for testing and comparing the performance of proposed algorithm. They are two and three objective problems with multi variables, shown in Table 5.2. First function is a two objective function that has a continuous convex Pareto optimal front shown in Figure 5.3. The second function is a two objective test function with a discontinuous convex Pareto front shown in Figure 5.6. The third test function is a three objective function that is shown in Figure 5.9.

Table 5.2-Test functions

Test function 1:	Test function 2:	Test function 3:
$g(x_1,\ldots, x_{30}) = 1+9(\sum_{i=1}^{30} x_i)/(n-1)$	$g(x_1,\ldots, x_{30}) = 1+9(\sum_{i=1}^{30} x_i)/(n-1)$	$f_1(x)=(1+x_3)\cos(x_1\pi/2)\cos(x_2\pi/2)$
$h(f_1,g) = 1-\sqrt{(f_1/g)}$	$h(f_1,g) = 1-\sqrt{(f_1/g)}-(f_1/g)\sin(10\pi f_1)$	$f_2(x)=(1+x_3)\cos(x_1\pi/2)\sin(x_2\pi/2)$
$f_1(x_1)= x_1$	$f_1(x_1)= x_1$	
$f_2(x) = g(x_1,\ldots, x_{30}).\ h(f_1,g)$	$f_2(x) = g(x_1,\ldots, x_{30}).\ h(f_1,g)+1$	$f_3(x)=(1+x_3)\sin(x_1\pi/2)$
$x_i \in [0\ 1]$	$x_i \in [0\ 1]$	$x_i \in [0\ 1]$

5.11 Experimental Results and Comparisons

In this section the results of implementation of MLMOPSO on three test functions presented in previous section are compared with two other algorithms. The first algorithm is Sigma method in (Mostaghim and Teich 2003) and the second method, called One-Leader method here, which was introduced in (Bartz-Beielstein et al. 2003). As described in Section 6, these two mentioned methods were modified with proposed archiving technique. Same swarm size, maximum *archive* size and parameters were used for each to have a justified comparison between two common strategy of choosing leaders in the swarm and the new proposed strategy. In order to have a justified comparison without the effect of stopping criteria, 40 runs performed for each algorithm on each test function while the stopping criteria was deactivated and all methods reached 100 iterations to stop. The worst and best results in the following tables are based on 40 runs of each algorithm on each test function for 100 iterations. However, in order to analyze the efficiency of each algorithm, it was applied on each test function with activated stopping criteria for 40 runs and the minimum iteration that they stopped was taken for efficiency analysis. The results that are presented for minimum iteration are based on the minimum iteration at which each algorithm obtained a reasonable solution set with having the stopping criteria.

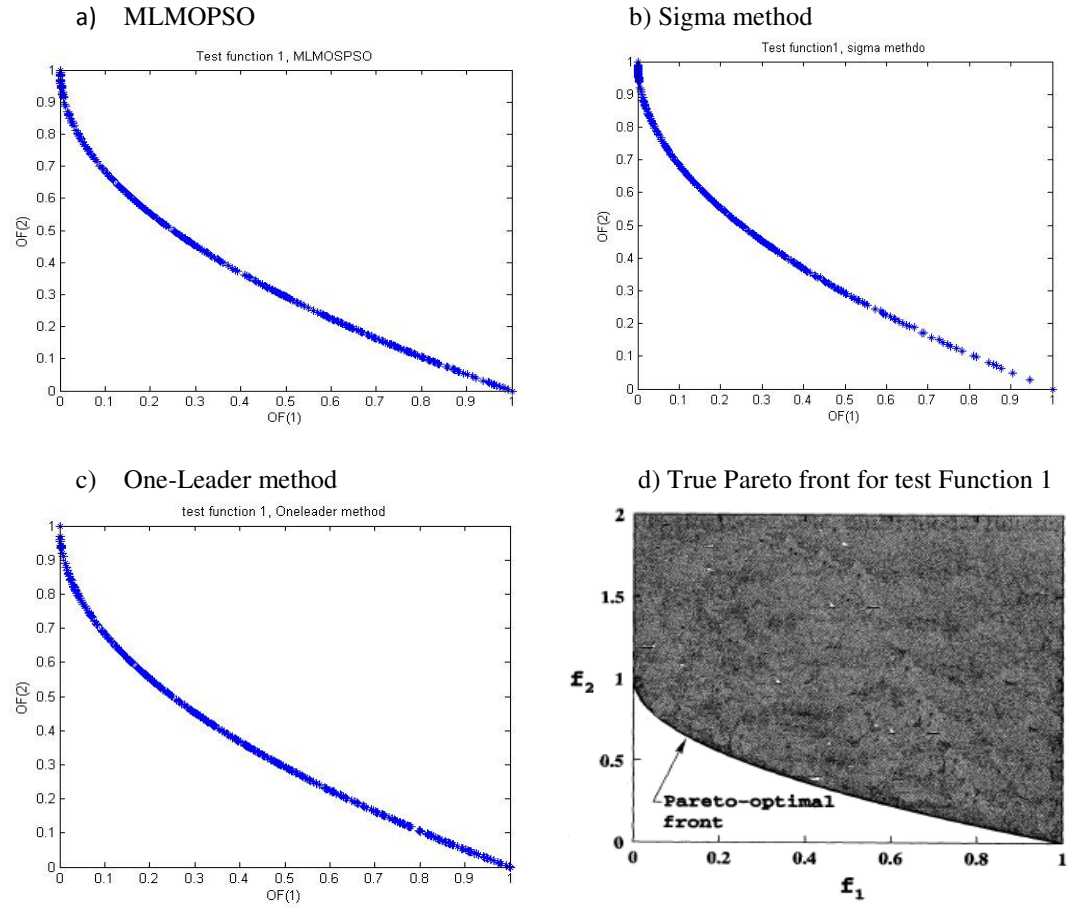


Figure 5.3- Best Pareto front of test function 1 obtained from 40 runs of a) MLMOPSO method b) Sigma method c) One-Leader method. d) The true Pareto front (Deb et al. 2002)

Figure 5.3 shows the best results obtained using each method from 40 runs. As it can be seen in this figure, although all methods can locate a Pareto optimal front close to the true Pareto front the MLMOPSO method obtains a more uniform solution set. Table 5.3 presents the metrics for each method. As it can be seen, all methods have almost the same Spread which means they are very good in locating the extremes. However, MLMOPSO method obtains a more uniform solution set. Also MLMOPSO is better in terms of S-metric which indicates better diversity and closeness to the true Pareto solution set. MLMOPSO according to the weakly dominancy metric shows an absolute superiority. One-Leader method shows a better performance over the Sigma method in terms of all metrics presented in Table 5.3 for the best solution set obtained from 40 different runs. Also Table 5.4 4 presents the percentage of solutions by the method in the column

which are weakly dominated by the method in the row. Zero shows that all the solutions are the same. And any number between (0 1] shows the percentage of weakly dominated solutions. As it is presented in Table 5.4 MLMOPSO outperform the other algorithms in terms of this performance metric.

Another important quality of a population based optimization algorithm is consistency of obtaining a good solution. MLMOPSO showed very consistence capability in finding a Pareto front close to the true Pareto front in all 40 runs. It can be inferred by comparing the performance metrics of the worst result obtained by MLMOPSO method, presented in Table 5.5 and the best result, presented in Table 5.3, which are very close. One-Leader method has the least consistency since it converged to only three Pareto sets with good quality out of 40 runs. Again comparison of its best and worst result out of 40 runs confirms less consistency in these methods.

Table 5.3-Comparison of results for Function 1 for the best Pareto set obtained by each in 40 runs

Indicator	MLMOPSO	Sigma	One-Leader
Spread	1.4142	1.4142	1.4743
Non-uniformity	1.0650	1.5435	1.1787
S-metric	1.2681e+004	7.7813e+003	9.3262e+003

Table 5.4-Comparison of best results of each algorithm for Function 1 in terms of dominancy indicator

Method	MLMOPSO	Sigma	One-Leader
MLMOPSO	0	1	1
Sigma	0.9492	0	0.9576
One-Leader	0.9234	0.8231	0

Table 5.5-Comparison of results for Function 1 for the worst Pareto set obtained byeachin 40 runs

Indicator	MLMOPSO	Sigma	One-Leader
Spread	1.3994	1.3073	1.4142
Non-uniformity	1.4098	2.0542	2.0743
S-metric	8.7066e+003	5.3408e+003	250.9647

Table 5.6-Comparison of results for Function 1 for the Pareto set obtained byeachat minimum iteration out of 40 runs

Indicator	MLMOPSO at iteration 21	Sigmaat iteration57	One-Leader at iteration35
Spread	1.4132	1.3151	1.4142
Non-uniformity	1.18	1.8638	1.9247
S-metric	1.0195e+004	5.8155e+003	1.8417e+003

a) MLMOPSO b) Sigma method c) One-Leader method

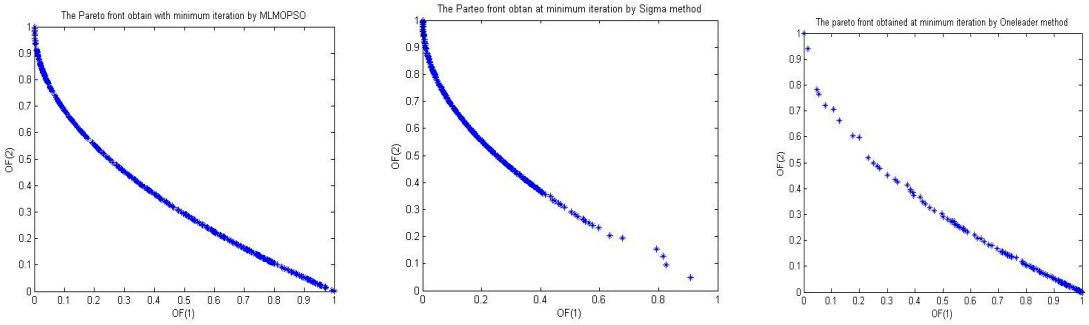


Figure 5.4- The Pareto front of test function 2obtained at minimum generation by a) MLMOPSO method b) Sigma method c) One-Leader

The Pareto fronts obtained at the minimum iteration by three methods are presented in Figure 5.4 and their corresponding performance metrics are presented in Table 5.6. From these results it can easily be inferred that for Sigma method and One-Leader method it is quite probable to converge to a solution set with less quality and diversity. Also, when the algorithm stops before reaching the upper limit of iterations it means there is no significant improvement in the *archive* or non-dominated set. Their performance is not as consistent as the performance of MLMOPSO

method and they need more iteration to converge to a solution set with good quality. MLMPSO method can obtain a high quality solution consistently even in few number of iterations as it is shown in Table 5.6. The average number of iterations that MLMOPSO converged to a good quality Pareto set is 28, which proves its efficiency in convergence. The average number of function evaluation (NOFE) for the result presented in Figure 5.4 is 2800. The number of function evaluations is much less than 5000 introduced by Sierra and CoelloCoello (2006)as the optimum NOFE for a very effective MOPSO algorithm.

a) Pareto front obtained in this work (2003)

b) Pareto front obtained by Mostaghim and Teich

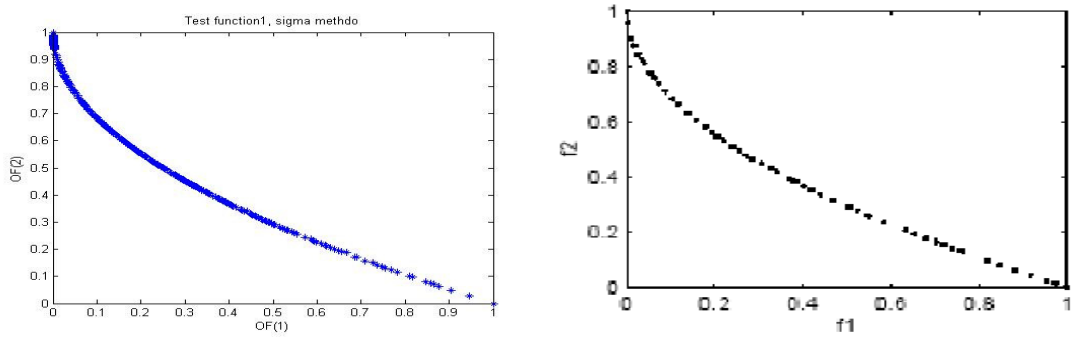
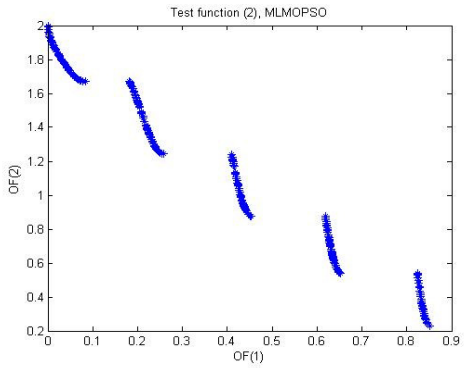


Figure 5.5- a) Pareto front of test function 1 obtained by a) Sigma method modified in this work b) Sigma method implemented in Mostaghim and Teich (2003)

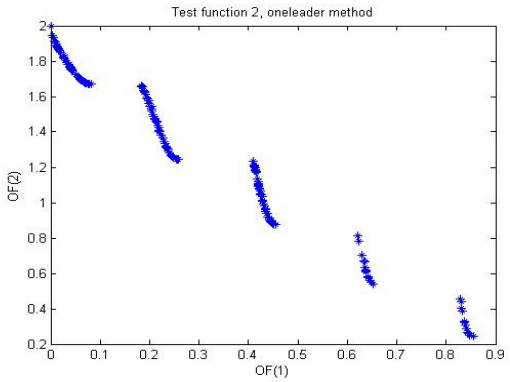
Mostaghim and Teich(2003) used the sigma method with 300 particles in the swarm and terminated the algorithm at 100 iterations. Their original algorithm was modified in this work as mentioned in section 5.6 and 5.8 with the archiving technique and initialization method. As shown in Figure 5.5 although they used higher number of particles, 100 iterations and applied mutation and cross over operator, better quality of the Pareto front obtained for Function 1 in this work, shows that the new archiving technique presented in this work maintains diversity and quality in the final solution set.

The result of the best implementation out of 40 runs for the test function 2 is presented in Figure 5.6. Although all methods converge to a Pareto front very close to the true Pareto front, MLMOPSO method obtains a more uniform solution set.

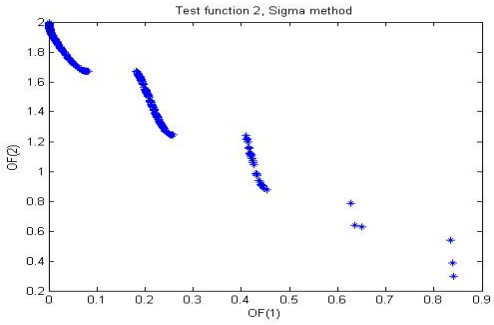
a) MLMOPSO method



b) Sigma method



c) One-Leader method



d) the true Pareto front of test function 2

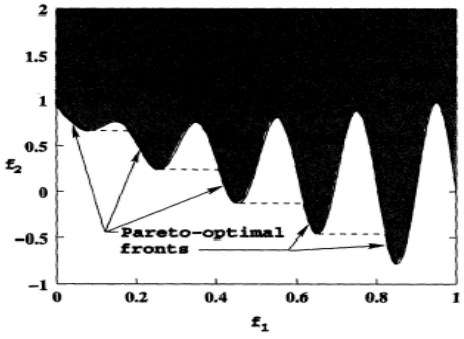


Figure 5.6- Best Pareto front of test function 2 obtained from 40 runs of a) MLMOPSO method b) Sigma method c) One-Leader method. d) The true Pareto front (Deb et al. 2002)

Table 5.7 presents the metrics for each method. It shows that MLMOPSO can find a more uniform Pareto front and also can locate the extremes of functions; metrics shown in the Table 5.7also confirm this result. Sigma method has more consistent performance than the One-Leader method. MLMOPSO method is the one with highest probability to converge to a good quality Pareto front. This can be inferred from the metrics of the best and worst solution obtained out of 40 runs. Comparison of the performance metrics of the worst result obtained by MLMOPSO

method, presented in Table 5.9, and the best result, presented in Table 5.7, confirms that MLMOPSO method is very consistent in terms of converging to good quality solutions. Table 5.8 presents the dominancy performance metric which is the percentage of solutions by the method in the column which are weakly dominated by the method in the row. As it is presented in Table 5.8 MLMOPSO has better performance comparing to the other methods in terms of this metric.

Table 5.7- Comparison of results for Function 2 for the best Pareto set obtained by each in 40 runs

Indicator	MLMOPSO	Sigma	One-Leader
Spread	1.9669	1.9001	1.9505
Non-uniformity	1.1962	1.9440	1.3562
S-metric	9.3206e+003	6.5091e+003	3.8462e+003

Table 5.8- Comparison of best results for Function 2 in terms of dominancy indicator

Method	MLMOPSO	Sigma	One-Leader
MLMOPSO	0	0.9901	0.9868
Sigma	0.9561	0	0.9503
One-Leader	0.8458	0.9005	1

Table 5.9- Comparison of results for Function 2 for the worst Pareto set obtained by each in 40 runs

Indicator	MLMOPSO	Sigma	One-Leader
Spread	1.9352	1.5547	1.8433
Non-uniformity	1.7962	2.5354	2.7716
S-metric	8.6126e+003	2.4654e+003	903.5142

Table 5.10- Comparison of results for Function 2 for the Pareto set obtained by each at minimum iteration out of 40 runs

Indicator	MLMOPSO at iteration20	Sigma at iteration46	One-Leader at iteration31
Spread	1.9591	1.5547	1.9490
Non-uniformity	1.7962	2.0098	1.857
S-metric	9.0951e+003	4.4151e+003	903.5142

a) MLMOPSO b) Sigma method c) One-Leader method

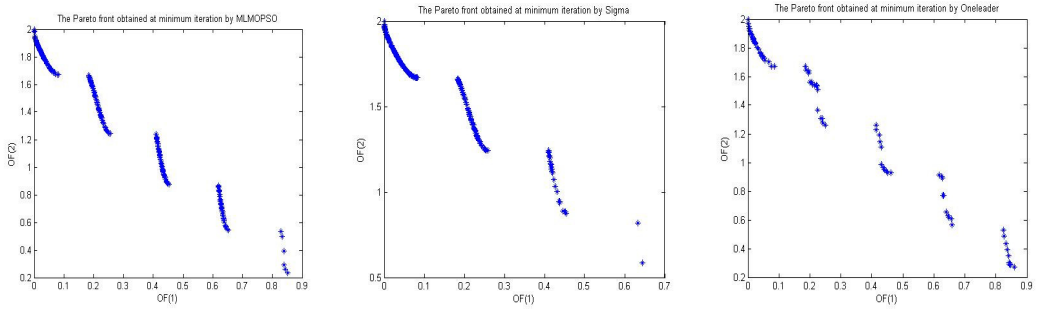


Figure 5.7- The Pareto front of test Function 2 obtained at minimum generation by a) MLMOPSO method b) Sigma method c) One-Leader

The Pareto fronts obtained at the minimum iteration by three methods are presented in Figure 5.7 and their corresponding performance metrics are presented in Table 5.10. These results confirm the superiority of MLMOPSO in terms of efficiency and convergence quality as discussed for previous test functions. Also, the average NOFE for the result presented in Figure 5.7 was 3600 confirms the efficiency of the new algorithm.

a) Pareto front obtained in this work (2003)

b) Pareto front obtained by Mostaghim and Teich

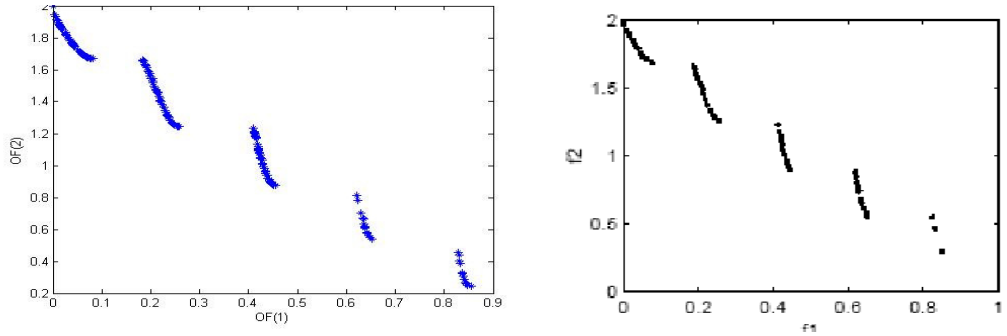


Figure 5.8- a) Pareto front of test function 2 obtained by a) Sigma method modified in this work b) Sigma method implemented in (Mostaghim and Teich 2003)

Again the impact of new modifications, archiving technique and initialization method presented in section 7, is established by comparing the result obtained from implementation of Sigma method by its original author and implementation of modified sigma method in this work, Figure 5.8.

Function 3 is a three objective concave function and was selected to test the performance of algorithms in handling problems with higher number of objectives. The true Pareto front of this function is quarter of a sphere. The error of solutions obtained using algorithms is calculated and presented as a histogram to show the closeness of the obtained Pareto front to the true Pareto front.

$$Error = 1 - f_1(x_i)^2 - f_2(x_i)^2 - f_3(x_i)^2 \quad (5.13)$$

Where f_i is the value of objective functions calculated at x_i.

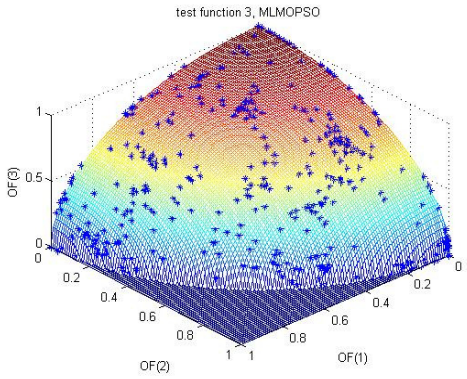
The results presented in Figure 5.9 through Figure 5.10 and table 5.11 through Table 5.14, confirm the superiority of MLMOPSO in handling multi-objective problems and obtaining a diverse Pareto front close to the true Pareto front. MLMOPSO is very consistent in finding solutions with high quality and very small error from true Pareto front even in very few iteration

numbers. The average number of function evaluation for MLMOPSO was 39000 in 40 different runs of the algorithm. MLMOPSO can be concluded as a consistent method by comparing the metrics calculated for its worst and best solutions. MLMOPSO converged to a wide diverse solution with very small error for all runs. Sigma method also is consistent in converging to good solutions but with higher error relative to MLMOPSO method. However, One-Leader method is not consistent in obtaining good solutions and converges to the final solution sets with higher error and less quality.

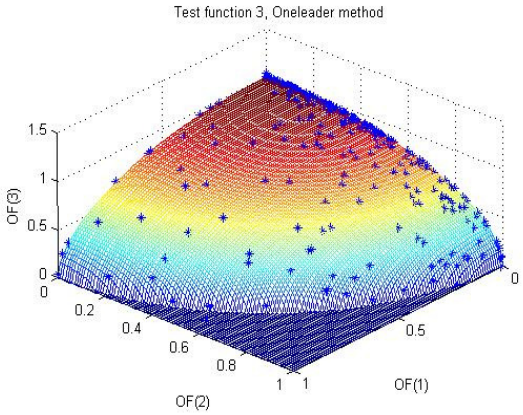
The comparison of diversity of solutions, Figure 5.12, and the histogram of the error, Figure 5.10 c and d, obtained from the implementations of Sigma method in the present work and the work by Mostaghim and Teich(2003) implies the capability of new archiving method in maintaining the solutions with better quality in the *archive*. The larger spread is more desirable for a solution set but spread for the test function that their true Pareto front are known cannot exceed a certain value because of the definition of the spread metric. Therefore, the spread higher than 1.7321 for function 3, 1.97 for function 2 and 1.4142 for function 1 indicate an error in locating the extremes of objective functions.

As stated by Deb et al. (2002) performance of NSGA II on this problem with 200 generations and a population size of 100 was not satisfactory due to poor distribution of final solutions. Results obtained by NSGAII for three runs on Function 3, presented in Figure 10, show inaccuracy of the obtained results comparing to the true Pareto front. This demonstrates the superiority of MLMOPSO over NSGAII in terms of NOFE and accuracy of final Pareto solutions. In addition, final solutions obtained by NSGAII were dependent on initial populations of the algorithm which reduces the consistency and reliability of the algorithm (Deb et al., 2002). However, final solutions obtained with the proposed algorithm are less dependent on initial populations and MLMOPSO is very consistent in finding final solutions with very close to the true Pareto front.

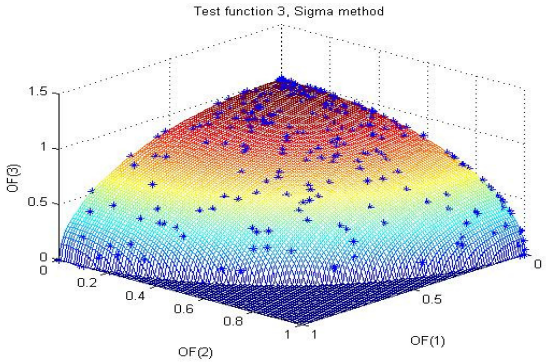
a) MLMOPSO



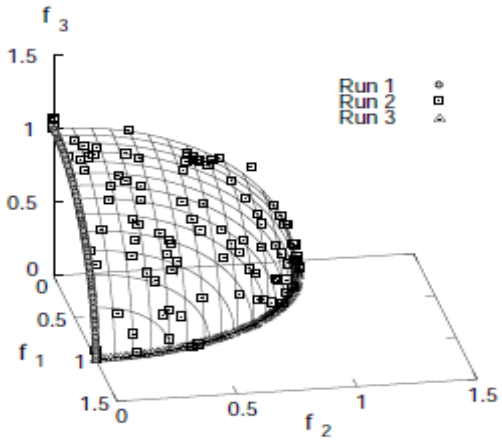
b) One leader method



c) Sigma method



d) NSGAII [16]



e) True Pareto front

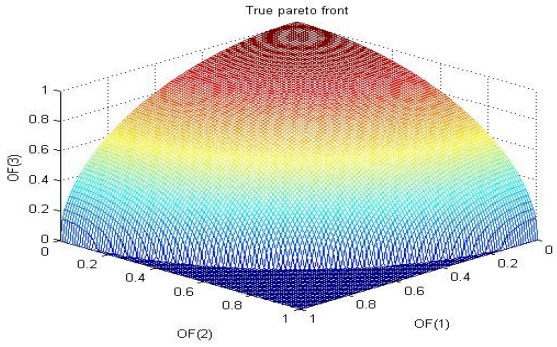
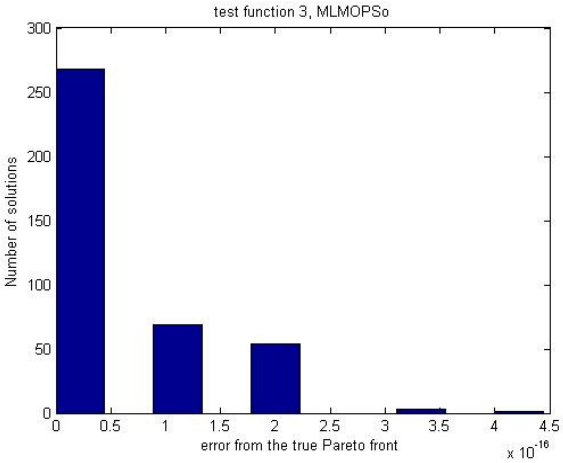


Figure 5.9- Best Pareto front of test function 3 obtained from 40 runs of a) MLMOPSO method b) One-Leader method c) Sigma method. d) The true Pareto front (Deb, Thiele et al. 2002)

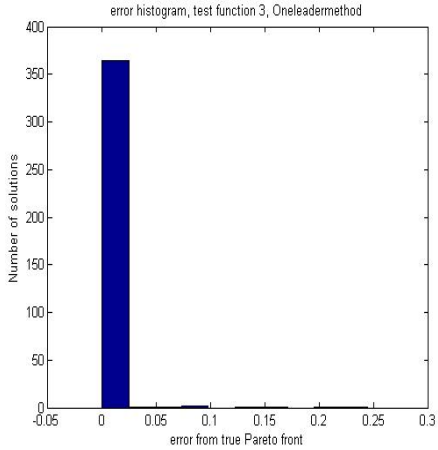
The comparison of the results obtained from implementation of the algorithms using proposed archiving technique with implementation of the same algorithms in previous works on the same test functions was shown for each test function. Much better results with small errors

prove that the proposed archiving technique is capable of maintaining solutions with good quality and diversity in the *archive* and this, results in convergence to better solution sets.

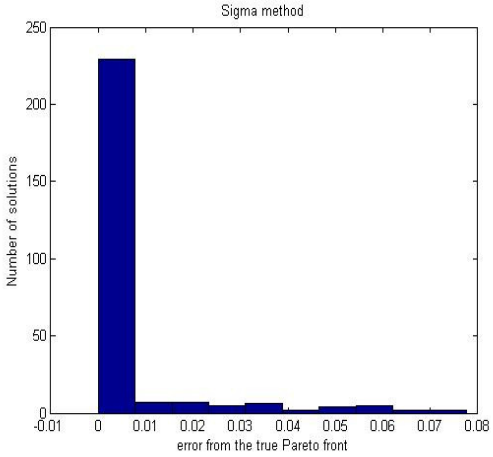
a) MLMOPSO



b) One-leader method



c) Sigma method



d) original work by Mostaghim and Teich (2003)

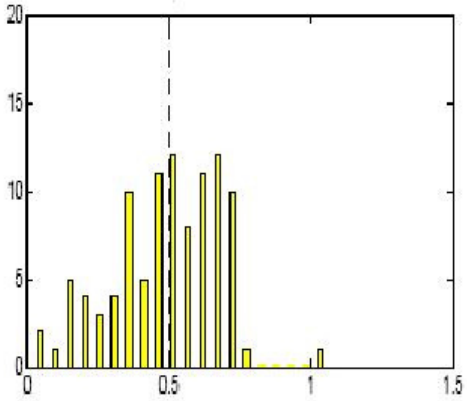


Figure 5.10- Histogram of error from the true Pareto front of function 3 obtained by a) MLMOPSO method b) One-Leader method c) Sigma method d) original work by Mostaghim and Teich(2003).

Table 5.11- Comparison of results for Function 3 for the best Pareto set obtained by each in 40 runs

indicator	MLMOPSO	Sigma	One-Leader
Spread	1.7321	1.7589	2.9790
Non-uniformity	1.0959	1.8135	1.6677
S-metric	4.5602e+004	3. 1039e+004	4.0804e+003

Table 5.12- Comparison of best results for Function 3 in terms of dominancy indicator

Method	MLMOPSO	Sigma	One-Leader
MLMOPSO	1	1	1
Sigma	0.9517	1	0.9665
One-Leader	0.9763	0.9868	1

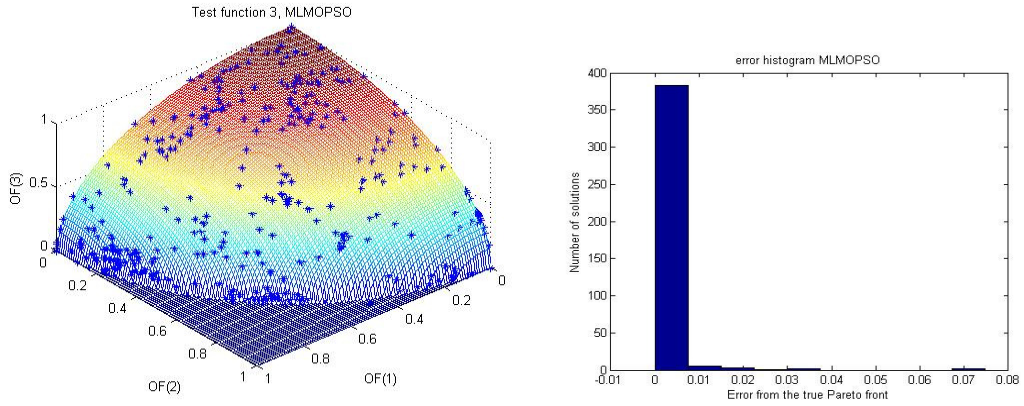
Table 5.13- Comparison of results for Function (3) for the worst Pareto set obtained by each in 40 runs

Indicator	MLMOPSO	Sigma	One-Leader
Spread	1.7321	1.7429	2.6137
Non-uniformity	1.1124	2.1054	4.18573
S-metric	4.0679e+004	3.7523e+003	653.3527

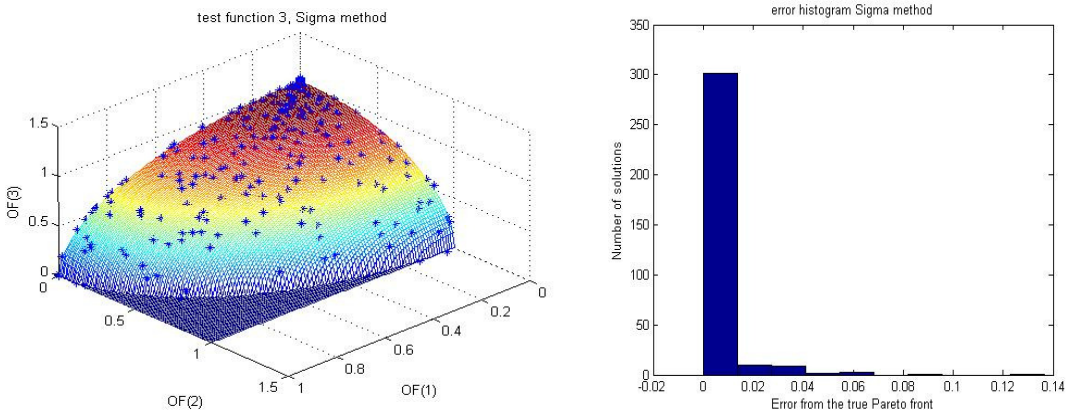
Table 5.14- Comparison of results for Function 3 for the Pareto set obtained by each at minimum iteration out of 40 runs

Indicator	MLMOPSO at iteration 23	Sigma at iteration 31	One-Leader at iteration 33
Spread	1.7321	1.5547	1.9490
Non-uniformity	1.1001	2.4818	1.857
S-metric	4.1979e+004	3.0039e+004	8.9828e+002

a) MLMOPSO



b) Sigma method



c) One-Leader method

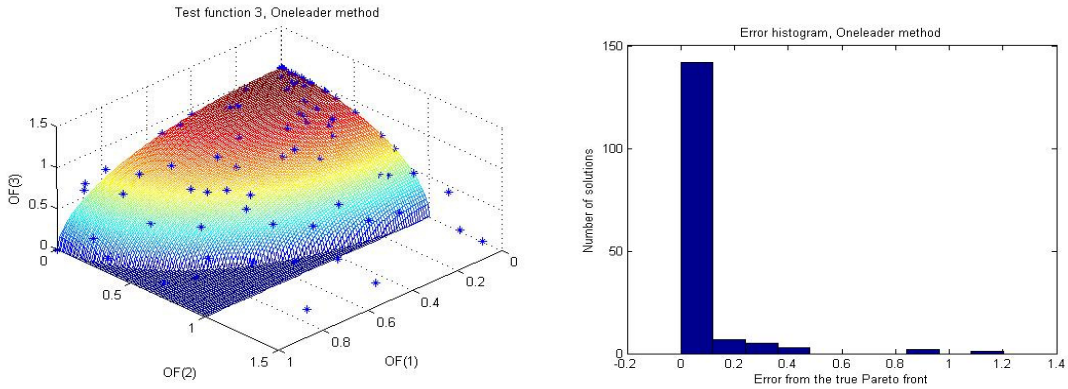


Figure 5.11- The Pareto front and error histogram of test Function 3 obtained at minimum generation by a) MLMOPSO method b) Sigma method c) One-Leader

a) Pareto front obtained in this workb) Pareto front obtained by Mostaghim and Teich

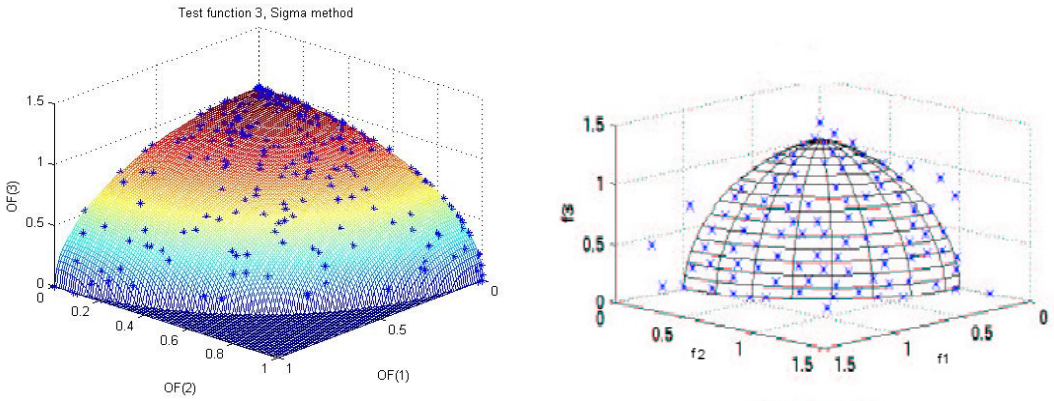


Figure 5.12- a) Pareto front of test function 3 obtained by a) Sigma method modified in this work b) Sigma method implemented in (Mostaghim and Teich 2003)

5.12 Implications of Results Obtained on Unconstrained Test Functions

As the experiments show, the performance of the MLMOPSO is better than the other two methods that use existing concepts for updating position of particles and selecting leaders in the swarm. MLMOPSO shows a very consistent performance for all three test functions and converges to final Pareto optimal sets with high quality and diversity with small error from the true Pareto front. This implies that this method is more reliable for handling real problems whena priori knowledge about their true Pareto front and functionality of the objective functions are not available.

According to the performance metrics, for test Function 1 the Sigma method had the better performance over the One-Leader method and according to test Function 2 One-Leader method is better. However, the One-Leader method was less consistent than the sigma method and the presented result was the best out of40 runs. The large improvement in the error of modified Sigma method shows the positive effect of proposed archiving technique in maintaining solutions

with high quality. However, much smaller amount of errors obtained by MLMOPSO shows the capability of Equation 5.4 in the proposed MLMOPSO method.

As Sierra and Coello Coello(2006) mentioned MOPSO algorithms are very effective in term of data structures, memory management. Therefore, making these algorithms more complex and intricate by adding complexity such as different operators and features inserting computational burden into algorithm is not reasonable. The new proposed PSO based algorithm is very effective in term of converging to very good quality solutions with less than 5000 ANOFE (average number of function evaluation) without a need of applying any extra operator and complexity such as clustering analysis or incorporating sub populations to the algorithm.

5.13 Modification of MLMOPSO for Constrained Mixed Integer Optimization

Global optimization has been in demand for decades for nonconvex mixed integer decision making problems. This type of mathematical programming handles nonlinear objective functions and constraints with a decision variable space containing both continuous and discrete variables. Several researchers have developed a generic optimization algorithm capable of locating the global optima accurately with least computational burden. Some of well-known methods are: a branch and bound technique(Grossmann and Sargent 1979); an outer-approximation (OA)/equality relaxation algorithm(Kocis and Grossmann 1988); outer-approximation (OA) based algorithm(Diwekar et al. 1992); a branch and reduce algorithm based on a sequential underestimation of convex region(Ryoo and Sahinidis 1995); a global optimization algorithm for nonconvex generalized disjunctive programming and (Lee and Grossmann 2001); a genetic evolutionary algorithm (Costa and Oliveira 2001); a genetic algorithm for distillation sequencing (Leboreiro and Acevedo 2004); and a differential evolution algorithm (Babu et al. 2005).

Although several researchers have improved the success rate and efficiency of PSO (Afshinmanesh et al. 2005; Liao et al. 2007; Xu and Liu 2009; Yiqing et al. 2007), the efficiency

of PSO algorithm in handling constrained MINLP problems is not as good as the original PSO. In this section a new method is presented for handling constraints in nonconvex MINLP problems which is the main reason of inefficiency. This method uses a highly effective multi-leader version of PSO proposed by the author in handling multi-objective problems. Also, the method for handling binary variables presented by (Khanesar, et al. 2007) is utilized to adapt the algorithm for handling mixed integer variables. The application of the proposed algorithm over a variety of test problems reported in previous works shows a superior performance in locating the global optima and inefficiency.

As was mentioned before, the original PSO is not suited for handling optimization problems with a mixed integer decision area. Due to the prevailing role of mixed integer and combinatorial decision creating challenges in the realm of engineering, PSO has been modified for handling this type of problems by various researchers. For the first time Kennedy and Eberhart (1997) modified the original PSO algorithm; however other researchers presented several modifications. In the subsequent sections the original modified method from Kennedy and Eberhart (1997) is presented and a modification on that developed by Khanesar et al. (2007) is explained and used for the purpose of this work.

5.13.1 Mixed Integer Handling Methods

5.13.1.1 The Method by Kennedy and Eberhart

Kennedy and Eberhart(1997) redefined the meaning of velocity as the change in the probabilities of being one or zero for each discrete variable. While discrete variables move in a space restricted to zero and one the velocity corresponding to each discrete variable represents the probability of being one for that variable. In other word if $v_i = 0.2$ it means the chance of being one for x_i is twenty percent. Therefore in this method the difference between previous position of

x_i and its local best can be calculated as 1, 0 or -1 to reasonably weight the change in probability of v_i at the next step.

The probability of being one is calculated as the velocity of each variable using the following formula.

$$v^i_{j,t+1} = wv^i_{j,t} + c_1r_1(G^i_{j,t} - x^i_{j,,t}) + c_2r_2(P^i_{j,t} - x^i_{j,,t}) \quad (5.14)$$

Therefore the basic formula for updating the velocity remains unchanged from the continuous PSO. The only difference is that P_i, G_i and xi are integer variables. Since v_i is a probability it must be constrained in the interval of [0 1]. A log-sigmoid transformation is used to transform the velocity to a probability. Then the corresponding position is updated using the following rule:

$$\left.\begin{array}{l} \textit{If } (rand < sigmoid\ (v^i_{j,t})) \textit{ then } x_i = 1 \\ \textit{Else } x_i = 0 \end{array}\right\} \quad (5.15)$$

Where: $sigmoid = 1/(1+exp(-x))$

and rand is a uniform distributed random number.

As with the continuous particle swarm optimization, there is a limit on the velocity and v_i is restricted as $|v^i_{j,t}| < v_{max}$ Unlike the continuous particle swarm optimization that a higher velocity enhances the exploration, in the discrete particle swarm optimization using Kennedy and Eberhart method, a higher velocity provides a lower mutation on the discrete variables and thus, the exploration capability is lower.

However, there is another substantial difference between the original PSO and the revised discrete version using Kennedy and Eberhart method. As Kennedy and Eberhart (1997) stated, although $v^i_{j,t}$ represents a probability threshold, changes in the velocity may cause a change in the position. Unlike the original, in Kennedy and Eberhart method the trajectory is a probabilistic model. The velocity on a certain discrete dimension of a particle is the probability that the corresponding variable changes; therefore even if based on v_i the discrete variable should remain

fixed, the position of the particle may change due to the stochastic characteristic of the v_i Kennedy and Eberhart (1997).

The rate of change for a bit (a discrete variable of a particle) of a particle is a non-directional probabilistic change. Probability of being one for a bit is *sigmoid* ($v^i_{j,t}$) and the probability for it to change to zero is *1 –sigmoid* ($v^i_{j,t}$). Therefore, the probability of change for that particular discrete variable would be *sigmoid* ($v^i_{j,t}$)*(*1 –sigmoid* ($v^i_{j,t}$)).

In contrary to original continuous particle swarm, using this method for handling discrete variables makes the position of each particle to be ephemeral. A particular particle with a specific value of $v^i_{j,t}$ may have different position at every iteration due to stochastic character of its trajectory.

5.13.1.2 Method by Ahmadie Khansar et al.

As stated by (Khanesar et al. 2007)it is not just the interpretation of velocity that differs from the original PSO in the binary version proposed by Kennedy and Eberhart(1997), but also how the velocity, inertia weigh and velocity upper and lower limit impact the search inversely comparing to those in the continuous version.

In the continuous PSO, a larger velocity enhances the search even if a good solution has been found. However, in the binary version of PSO, based on the Kennedy and Eberhart method (1997), smaller values of velocities promote the search. A velocity equal to zero turns the search to a pure random search. Therefore, too large a velocity or inertia limits the exploration capability of the algorithm. Thus, Eberhart and Kennedy (1997) suggested $v_{max} = 6$ to be used. It is also argued that for the value of the inertia less than one, $w < 1$ binary version of PSO does not converge. The values of $-1 < w < 1$ make the velocity zero over the time. Negative values of the inertia reduce the velocity over time and increase the probability of being zero. This phenomenon is significant in the optimization of problems with several binary variables. Therefore, the choice

of inertia parameter seems to be a challenge. Several methods have been proposed to solve this problem by removing the inertia parameter or choosing it randomly during the optimization process. However none of these choices seem to be reasonable because the inertia parameter contains valuable information about previously found positions. In addition, the equation that is used to update the position of particles in Kennedy and Eberhart method is not dependent on the current position of particles and does not carry any information of it. In the original continuous PSO the velocity vector determine the movement and direction of the particle path on the datum of the current position of it (Khanesar et al. 2007).

In *Ahmadie Khansar* method which is chosen for this study, the major difference is the interpretation of velocity. Here the velocity is the probability of being changed for a specific bit of a particle and each bit has two velocities; v_{ij}^{1} and v_{ij}^{0} which are the probability of change to one and zero respectively for a bit. Since there is an inertia term in the updating equation, introduced later, these velocities are not complementary. Therefore the probability of change for the j^{th} bit of the i^{th} particle is:

$$If\ x_{ij}=0\ then\ v_{ij}^{c}=v_{ij}^{1}$$

$$Elseifx_{ij}=1\ then\ v_{ij}^{c}=v_{ij}^{0} \tag{5.16}$$

The position of each is updated based on its current position using either velocity of one or zero. The velocities for each bit are calculated using the following rules:

$$\left.\begin{array}{l} If\ P_{ij}=1\ then\ d_{ij,1}^{1}=c_1r_1\ and\ d_{ij,1}^{0}=-c_1r_1 \\ If\ P_{ij}=0\ then\ d_{ij,1}^{0}=c_1r_1\ and\ d_{ij,1}^{1}=-c_1r_1 \\ If\ G_{ij}=1\ then\ d_{ij,2}^{1}=c_2r_1\ and\ d_{ij,2}^{0}=-c_2r_2 \\ If\ G_{ij}=1\ then\ d_{ij,2}^{1}=c_2r_2\ and\ d_{ij,2}^{1}=-c_2r_2 \end{array}\right\} \tag{5.17}$$

Where, $d_{ij,1}^{1}$and $d_{ij,1}^{0}$are two temporary values and $r_1$and r_2 are two random numbers. c_1, c_2 and w were fixed parameters of the algorithm.

$$v_{ij}^{1}=wv_{ij}^{1}+d_{ij,1}^{1}+d_{ij,2}^{1}$$

$$v_{ij}^{0} = w v_{ij}^{0} + d_{ij,1}^{0} + d_{ij,2}^{0} \quad (5.18)$$

In this method if the current value of the bit is zero and both global best G_{ij} and local best P_{ij} are zero, the velocity of zero increases and the velocity of one decreases so the probability of being one for that bit decreases. The position value of each bit is updated using the following formula after normalization using a log-sigmoid function:

$$\text{If sigmoid } (v^{c}_{ij}) > \text{random then } x_{ij}(t+1) = 1 - x_{ij}(t)$$

$$\text{If sigmoid } (v^{c}_{ij}) < \text{random then } x_{ij}(t+1) = x_{ij}(t) \quad (5.19)$$

The interpretations of the parameters in this method are the same as those in the original continuous PSO. The inertia parameter maintains the impact of the previous direction toward global or local best. The velocities here are the rate of change toward zero or one. Also as in the original PSO, larger velocities promote the exploration capability of the algorithm. Too large a value of velocity leads to a pure random search as well. It should be mentioned that in *Ahmadie Khanesar* method, the current position of particles is taken into account unlike the Kennedy and Eberhart method where only velocity plays the role in updating the position (Khanesar et al. 2007).

The ranges of acceleration and inertia coefficient which are used for updating binary variables are the same as the ranges for continuous variables. The reader is referred to the work by Bhattacharya and Samanta, (2010), Shi and Eberhart, (1998) and Trelea, I.C.(2003) for further information on selecting PSO parameters and convergence analysis.

5.13.2 Constraint Handling Method

Engineering decision making problems usually contain equality and inequality constraints. Searching for the global optimum location where nonlinear equality constraints are active is the most notorious challenge of optimization algorithms particularly stochastic methods. There are three categories of methods for handling constraints. The first category is based on generating

new solutions in the feasible region and methods in this category are called feasible search methods. These methods are for linear constraints which construct a convex feasible search area. However they can be modified for nonlinear constraints with a convex feasible area. These methods are not efficient for nonlinear problems and the assumption that the feasible search area is convex raises problem in handling nonconvex problems. Also they need a pre analysis to find the feasible search region.

The second category which are called repairing methods are based on several subpopulations which each use a subset of constraints and/or different methods for constraint handling. The feasible solution of each subset is communicated between the subpopulations until they have found a completely feasible solution. However, these methods reduce the efficiency of the algorithms significantly which not desirable in this work. The third category includes a method where they push the algorithm to minimize the amount of overall violation from the constraints as it progress simultaneously. The most prominent methods of this category are penalty methods, overall-violation minimization and constraint dominancy method. The penalty methods suffer from the ambiguity of the penalty factors. Even though the penalty method has been modified with several approaches, it is very case sensitive and reduces the success rate of the stochastic algorithm due to stochastic change of the behavior of the original objective functions. Also, when dealing with several dissimilar objectives the scaling of the penalty factors is a problem for this method.

Alternatively, constraint dominancy and overall-violation minimization methods reduce the amount of overall violations from all of the active constraints with two different approaches. In the constraint dominancy method, in order to find a non-dominated solution, the concept of dominancy is redefined based on the violation from constraints. This method allows solutions with a lower amount of violation from the constraints to be considered better. Basically, all feasible solutions dominate infeasible solutions even if the infeasible solutions are better in terms of the objective functions and among infeasible solutions those with the lower amount of

constraint violation dominate the others regardless of the objective functions. However, the concept of the dominancy between feasible solutions would remain to only be based on the value of the objective functions (Raquel and Naval, 2005). As the algorithm progresses the constraint-dominancy concept results in the selection of the more feasible solution as the leaders and local bests for the particles. Thus, this leads the swarm toward a part of the feasible region. However, if there are several local optima in a highly constrained problem with a nonconvex feasible region this method fails to locate the global optimum very often and results in a weak success rate for the algorithm. The reason for this phenomenon is that once one solution happens to be in or close to a part of the feasible region with a low number of active constraintsit dominates all other potential solutions and drags other particles in the swarm toward that particular region. This would cause the algorithm not to evolve toward the narrow feasible region with more active constraints which the global optimum is located.

Overall-violation minimization methods directly minimize the amount of constraint violation as an extra objective. This type of method is much more efficient in locating the global optima because it allows each trial solution to evolve and reduces the amount of infeasibility (violation from constraints). One solution that is better in terms of objective functions but has more overall violation may be a potential solution close to the global minima which if it evolves, it become a feasible solution in the neighbor of the global solution. Yiqing et al. (2007) presented a strategy using a two nested PSO algorithm where the outer PSO optimizes the original objective functions and the second optimizer finds the dependent variables defined in equality constraints.For those constraints that are not solvable explicitly; some dependent variables are defined based on other independent variables so that the degree of freedom of the system is zero. The independent variables are determined by the outer PSO algorithm and the inner PSO determines the independent variables as the overall violation from constraint for that particular particle (trial solution) is minimized. The main deficiency of this strategy is that for the most of constrained problems it is not possible to solve dependent variables explicitly based on the independent

variables. Therefore, the second inner optimization must be performed for each particle (trial solution) and this reduces the efficiency of the algorithm drastically. Also each specific problem should be reformulated to define the dependent variables.

In this work, the idea of minimizing an extra objective as the overall violation is used. However, this extra objective is minimized simultaneously along with other objectives to avoid the second inner optimization loop. As the algorithm progresses a non-dominated set is produced to lead the other particles. This is due to the existence of more than one objective. The non-dominated set contains the particles with the best original objective functions as well as the particles with the lowest overall violation. Thus, the infeasible particles that are better in terms of the original objectives with the potential of being the feasible global optimum get the chance to evolve more and lead the other particles toward the global optimum, even though they are less feasible, compared to more feasible solutions. At the same time the amount of violation from all active constraints is being reduced. With a Pareto set of solutions with one objective as the overall violation, it is very probable to locate the feasible range even it is nonconvex and narrow. For a specific fraction of iterations (60% of maximum iterations in this study for non-adaptive MLMOPSO) this strategy is applied so that all the potential solutions evolve.Then in order to expel infeasible solutions and keep only the best feasible solution as the leader (global best) the non-dominancy concept is applied for constraint handling. This allows for further minimization of the overall violation if there is not a feasible solution found yet or to improve the quality of the minimum found so far. However, the key point of this method is usage of the proposed MLMOPSO algorithm which is very efficient and capable of handling multi-objective optimization problems. Table 5.15 summarizes the features of the MLMOPSO.

Table 5.15-Summary of the features incorporated in MLMOPSO algorithm

An external *archive*	To preserve all good quality non-dominated solutions obtained by the algorithm. To provide the possibility of selecting best solutions as the leaders in the swarm.
Fitness functions	To screen and rank the solutions based on the diversity and success rate indicators. To select the high rank solutions in the *archive* as leaders in the swarm to enable more diversity and wider search capability.
Information of several leaders in the swarm is used to guide each particle	To maintain more diversity and avoid clustering in some limited region. To explore more the search space even between non-dominated solutions by changing the vector of velocity toward different leaders. To increase exploring of the feasible regions around non-dominated solutions as algorithm proceeds toward the end.
Use of a novel equation for updating the position	To minimize the computation burden for incorporating the information of the leaders in the vector of velocity of each particle.
Calculation of the distance of the leaders and particles in the decision variable space	To increase the efficiency of the algorithm by making the particles follow the leaders with more similar characteristics.
Selection of local best using dominancy concept	To select a local best for each particle with the best quality in the swarm among possible previous positions.
Use of an *archive* based stopping criteria	To detect any improvement in the solution set using two different type of indicators. To avoid the computational expense imposed by using a pre-specified maximum iteration number.
Use of a new method introduced by Khanesar et al. (2007) for updating binary variables	This method improves capability of the algorithm to handle problems with many binary variables. Also velocity and acceleration parameters and inertia have the same interpretation and impact on the algorithm as the original PSO.
New two-phase Constraint handling method	This method minimizes an extra objective (the overall infeasibility of solutions) simultaneously in the phase one. This would help to locate the feasible region very efficiently and by switching to the phase two it continuous to further search the feasible region and improves the solutions. This method is independent of penalizing parameters and also is very effective for highly constrained nonconvex problems.

5.14 Experimental Results on Constrained Problems

The proposed algorithm (MLMOPSO) is applied on10 constrained problems which most of them are process synthesis problems obtained from (Kocis and Grossmann, 1988; Kocis and Grossmann, 1987; Ryoo and Sahinidis, 1995; Vinante and Valladares, 1985) to investigate its performance. The first two problems are used for evaluating the performance of the algorithm and comparing its result with the results obtained by other researchers using deterministic methods. Problems 3, 4 and 5 are used for the comparison of the proposed algorithm with other stochastic methods. Problems 6, 7 and 8, which are more complex and almost all algorithms struggle to find the global minimum, are used for further investigation of the algorithm. Also the proposed algorithm was tested on two constrained multi-objective problems, Problems 9 and 10, to demonstrate the capability of the algorithm in handling non convex multi-objective problems. For all the test problems the fraction of iteration before switching to constraint dominancy approach is 60%. The overall violation of the constraints is calculated as following;

$$Overall\ violation = \sum_{i=1}^{n} max\ (g_i, 0) + \sum_{j=1}^{n} max\ (abs\ (h_j), 0) \qquad (5.20)$$

Where; g and h represent inequality and equality constraints respectively.

Problem 1.This problem is a process synthesis problem taken from Kocis and Grossmann (1988) and also has been studied by Floudas et al. (1989) and Ryoo and Sahinidis(1995). MLMOPSO was applied on this problem for 20 runs with 50 particles in the swarm and the maximum iteration number was 50.

$$Minf(x, y) = 2x + y$$

$$s.t.\ 1.25 - x_2 - y \leq 0$$

$$x + y \leq 1.6$$

$$0 \leq x \leq 1.6$$

$$y_1 \in \{0,1\}$$

The global optimum reported by Ryoo and Sahinidis(1995)is (x,y; f)=(0.5,1;2). It also has a local minimum as (x, y; f) = (1.118, 0; 2.236).For all implementations the parameter of MLMOPSO was fixed as; c_1=2.8, c_2=1.2, w=0.8, SIF=3. It converged to a minimum within 0.005% of the global minimum for %96 of runs.

Problem 2. This problem is also a two stage process system design problem taken from Kocis and Grossmann (1988), in the work of Floudas et al. (1989) and has been studied by (Ryoo and Sahinidis 1995). MLMOPSO was applied on this problem for 20 runs with 50 particles in the swarm and the maximum iteration number was 100.

$$Min\ 2x + 3x_2 + 1.5y_1 + 2y_2 - 0.5y_3$$

$$s.t.\ x + y_1 = 1.25$$

$$x_2^{1.5} + 1.5y_2 = 3$$

$$x_1 + y_1 \leq 1.6$$

$$1.333x_2 + y_2 \leq 3$$

$$-y_1 - y_2 + y_3 \leq 0$$

$$y_i \in \{0,1\}^3$$

$$0 \leq x \leq (1.12,\ 2.1)$$

The global optimum reported by Ryoo and Sahinidis(1995) is (x_1, x_2, y_1, y_2, y_3; f) = (1.118034, 1.310371, 0, 1, 1; 7.667180). For all implementations the parameter of MLMOPSO was fixed as; c_1=2.8, c2=1.2, w=0.8, SIF=3. It converged to a minimum within 0.03% of the global minimum for %98 of runs.

Problem 3. This problem originally was investigated by Kocis and Grossmann (1987). This problem also has been studied by other researchers (Cardoso et al. 1997; Costa and Oliveira 2001; Salcedo 1992; Yiqing et al. 2007).

$$Min\ f(x_1, x_2, y) = -y + 2x_1 + x_2$$

$s.t.\ x_1 - 2\exp(-x_2) = 0$

$-x_1 + x_2 + y \leq 0$

$0.5 \leq x_1 \leq 1.4$

$y_1 \in \{0,1\}$

The global optimum is $(x_1, x_2, y; f) = (1.3750, 0.3747, 1; 2.1247)$.

Problem 4. This problem originally was proposed by Kocis and Grossmann (1987) to select one between two reactors in order to minimize the objective function which is the cost of product. Other researchers also investigated this algorithm using different algorithms (Cardoso et al. 1997; Costa and Oliveira 2001; Salcedo 1992; Yiqing et al. 2007).

$Min\ f(x, y_1, y_2, v_1, v_2) = 7.5y_1 + 5.5y_2 + 7v_1 + 6v_2 + 5x$

$s.t.\ y_1 + y_2 = 1$

$z_1 = 0.9[1 - \exp(-0.5v_1)]\ x1$

$z_2 = 0.8[1 - \exp(-0.4v_2)]\ x2$

$z_1 + z_2 = 10$

$x_1 + x_2 = x$

$v_1 \leq 10y_1$

$v_2 \leq 10y_2$

$x_1 \leq 20y_1$

$x_2 \leq 20y_2$

$x_1, x_2, z_1, z_2, v_1, v_2 \geq 0$

$y_1, y_2 \in \{0,1\}^2$

The global optimum is: $(x, y_1, y_2, v_1, v_2; f) = (13.4258, 1, 0, 3.514237, 0; 99.239641)$.

Problem 5. This problem is a process synthesis problem taken from Ryoo and Sahinidis (1995) and originally was presented by (Yuan et al. 1988). Also it was studied by other researchers (Salcedo 1992; Cardoso et al. 1997; Babu and Angira 2002).

$$Min\ f(x_1, x_2, x_3, y_1, y_2, y_3, y_4) = (y_1 - 1)^2 + (y_2 - 2)^2 + (y_3 - 1)^2 - ln(y_4 + 1)$$

$$+(x_1 - 1)^2 + (x_2 - 2)^2 + (x_3 - 3)^2$$

$$s.t.$$

$$y_1 + y_2 + y_3 + x_1 + x_2 + x_3 \leq 5$$

$$y_3^2 + x_1^2 + x_2^2 + x_3^2 \leq 5.5$$

$$y_1 + x_1 \leq 1.2$$

$$y_2 + x_2 \leq 1.8$$

$$y_3 + x_3 \leq 2.5$$

$$y_4 + x_1 \leq 1.2$$

$$y_2^2 + x_2^2 \leq 1.64$$

$$y_3^2 + x_3^2 \leq 4.25$$

$$y_2^2 + x_2^3 \leq 4.64$$

$$x \geq 0$$

$$y_i \in \{0,1\}^4$$

The global optimum solution is (x_1, x_2, x_3, y_1, y_2, y_3, y_4; f) = (0.08333134703, 1.21815116152982, 1.428516675518183, 1, 0, 1, 1; 4.227981684108).The global minimum reported in literature is (x_1, x_2, x_3, y_1, y_2, y_3, y_4; f) = (0.2, 0.8, 1.907878, 1, 1, 0, 1; 4.579582).It should be mentioned that GA, M-SIMPSA and R-PSO converge to a local minimum for this problem as it was reported in (Yiqing, Xigang et al. 2007) and shown in Table 5.16.

For problems 3, 4 and 5, the parameter were fixed as; c_1=2.8, c2=1.2, w=0.8, SIF=2. The size of the swarm was 50 for Problems 3 and 4 and it was 70 for Problem 5. The maximum iteration number was 50 for all three and the algorithm was run 100 times for each. The result of the

implementations for these three problems are presented in Table 2 and are compared with the result presented in the work by Yiqing et al. (2007) in terms of number of function evaluation(NOFE) and percentage of run converged to the global optimum (NRC).

Table 5.16 - Results of NOFE/NRC for Problems 1–3 using different algorithms

Problem Number	GA[a]	M-SIMPSA	R-PSO[b]	MLMOPSO
3	13939/100	14440/100	3500/100	2500/100
4	22489/100	42295/100	4000/100	2500/100
5	102778/60	63751/97	30000/100	3500/100

[a]-Costa and Olivera (2001)
[b]-LuoYiqing et. al (2006)

Problem 6. This problem was taken from (Hock and Schittkowski 1981). Again this is a NLP problem with three nonlinear equality constraints. This problem also was studied by others (Mezura-Montes and Coello Coello 2005). The Iteration number of the proposed algorithm, MLMOPSO to obtain the optimum solutions was 200 and the number of particles was 50 in the swarm. SIF was set to 2 to maintain more diversity while extra objective for constraint handling is active.

$$Min\ f(x) = e^{x1x2x3x4x5}$$

s.t.

$$h1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$$

$$h_2(x) = x_2x_3 - 5x_4x_5 = 0$$

$h_3(x) = x^3_1 + x^3_2 + 1 = 0$

$-2.3 \leq x_i \leq 2.3 \ (i = 1, 2)$

$-3.2 \leq x_i \leq 3.2 \ (i = 3, 4, 5)$

The optimum solution is (x_1, x_2, x_3, x_4, x_5; f) = (-1.717143, 1.595709, 1.8277247, -0.763643, -0.763645; 0.05394). the results obtained on this problem out of 50 implementations are presented in Table 5.17 along with the result of 4 other algorithms presented in the work of Yiqing et al. (2007); improved PSO R-PSO required 30,000 iteration of the outer loop, stochastic ranking (SR) with 350,000 iteration numbers, adaptive segregational constraint handing evolutionary algorithm (ASCHEA) with 1,500,000 iteration numbers and simple multimembered evolution strategy (SMES) with 240,000 iteration numbers. These results can be found in the work by (Mezura-Montes and Coello Coello 2005). Again results for this nonlinear function obtained with a lot less number of iteration and the number of function evaluation using the proposed algorithm.

Table 5.17 - Results of Problems 6using different algorithm

Problem Number	Optimum	SR	ASCHEA	SEMS	R-PSO[b]	MLMOPSO
Best solution	0.0539489	0.05395	_	0.05398	0.05394	0.0539489
Mean solution	0.0539489	0.05700	_	0.16638	0.05398	0.05402
Worst solution	0.0539489	0.21691	_	0.16638	0.05398	0.05832

-converged to a non-optimal solution

[b]-LuoYiqinget.al (2006)

Problem 7. This problem was also taken from (Hock and Schittkowski 1981). It is a NLP problem with three non-linear equalities. This problem also was studied by (Mezura-Montes and Coello Coello 2005).

$Min\ f(x) = 3x_1 + 0.000001x^3_1 + 2x_2 + (0.000002/3)x^3_2$

s.t.

$g_1(x) = x_3 - x_4 - 0.55 \leq 0$

$g_2(x) = -x_3 + x_4 - 0.55 \leq 0$

$h_3(x) = 1000 \sin(-x_3 - 0.25) + 1000 \sin(-x_4 - 0.25) + 894.8 - x_1 = 0$

$h_4(x) = 1000 \sin(-x_3 - 0.25) + 1000 \sin(x_3 - x_4 - 0.25) + 894.8 - x_2 = 0$

$h_5(x) = 1000 \sin(-x_4 - 0.25) + 1000 \sin(x_4 - x_3 - 0.25) + 1294.8 - x_2 = 0$

$0 \leq xi \leq 1200\ (i = 1, 2)$

$-0.55 \leq x_i \leq 0.55\ (i = 3, 4)$

For this problem the optimum reported in the literature is x* = (679.9453, 1026.067, 0.1188764, −0.3962336), where f(x) = 5126.498 and the overall violation for this solution is 0.00037699, from (Yiqing et al. 2007). Although MLMOPSO could not reach completely feasible solution, the solution obtained was x* = (674.709, 1031.665, 0.1226, -0.39446), where f(x) = 5126.634with overall violation amount from all constraints equal to 0.0084619. This solution which is within 0.0027% of the global minimum was obtained by 300 iterations, 100 particles in the swarm and SIF=1.5.

This is a fact that different constraint handling methods result in different quality of solution for each constrained test problems (Koziet and Michalewicz 1999). Even though the proposed algorithm could not locate a completely feasible solution for the problem 7 it located a solution very close to the global solution with yet small amount of violation in 300 iterations which confirms its efficiency.

Problem 8. This problem was taken from (Kocis and Grossmann 1987). The superstructure of a small process plant shown in Fig 5.13 is formulated as a mixed integer nonlinear problem. The superstructure contains two alternative paths of producing B from A. the problem contains three binary variables and five continuous variables, two non-linear equations, three linear equations and two upper bounds.

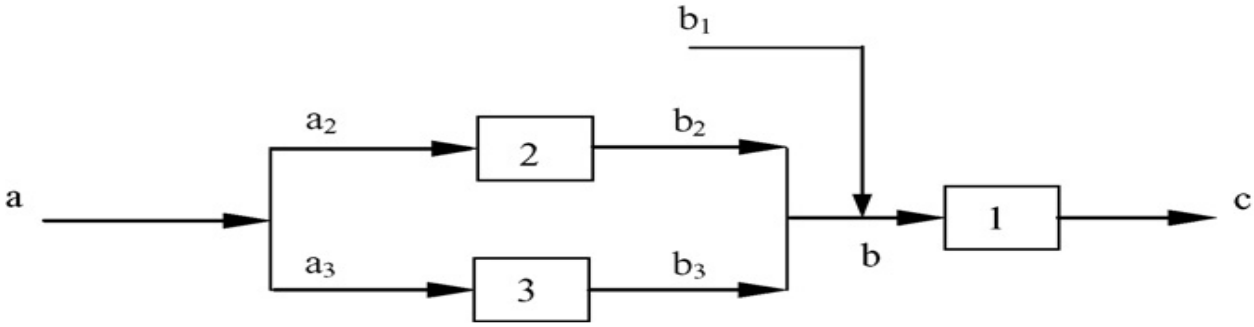


Figure 5.13- Superstructure of the process for problem 8

$Min\ f(y_1, y_2, y_3, b_1, b_2, b_3, a, c)$

$= 3.5y_1 + y_2 + 1.5y_3 + 7.0b_1 + b_2 + 1.2b_3 + 1.8a - 11c$

$b_2 - ln(1 + a_2) = 0$

$b_3 - 1.2\ ln(1 + a_3) = 0$

$c - 0.9b = 0$

$-b + b_1 + b_2 + b_3 = 0$

$a - a_2 - a_3 = 0$

$b - 5y_1 \leq 0$

$a_2 - 5y_2 \leq 0$

$a_3 - 5y_3 \leq 0$

$c \leq 1$

$b_2 \leq 5$

$y_i \in \{0,1\}^3$

$a, a_2, a_3, b, b_1, b_2, b_3, c \geq 0$

The global optimum is: (y1, y2, y3, b1, b2, b3, a, c; f) = (1, 0, 1, 0, 0, 1.11101576, 1.5240, 0.9999; −1.92309).For this problem the iteration number was 100 and the number of particles in the swarm was 50 and SIF was set to 1.5. The MLMOPSO converged to the global minimum 93% of times out of 100 runs. The number of function Evaluation (NOFE) was5000 which is much less comparing to NOFE required by improved PSO proposed by Yiqing et al. (2007). Unfortunately the NOFE required by improved PSO proposed by Yiqing et al. (2007)was not reported in their work.

To demonstrate capability of the algorithm on multi-objective problems the proposed algorithm were applied on the next two test problems and the Pareto solution obtained is presented for comparison with the true Pareto front calculated mathematically.

Problem 9. This problem was taken from (Deb, K. 2001).

$Min\ f_1(x) = x_1$

$f_2(x) = (1+x_2)/x_1$

s.t.

$g_1(x) = -x_2 - 9x_1 - 0.55 \leq -6$

$g_2(x) = x_2 - 9x_1 \leq -1$

$0.1 \leq x_1 \leq 1$

$0 \leq x_2 \leq 5$

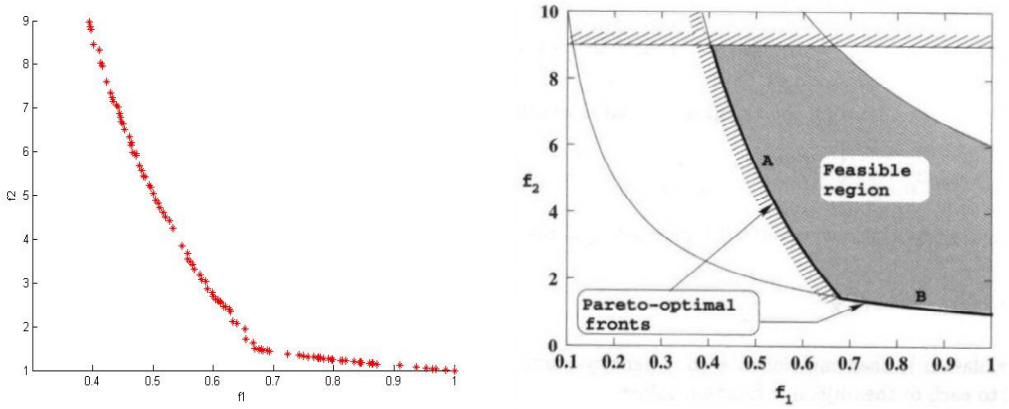


Figure 5.14- Comparison of the results obtained by MOMLPSO in 50 iterations and true Pareto front of Problem 9

The number of particles for this problem was 30 and the results shown in Figure 5.14 were obtained after 50 iteration of the algorithm and SIF was set to 2. Deb et al. (2202 b) applied NSGAII on this problem and obtained the Pareto front after 500 generation of the algorithm with population size of 100. This again confirms superiority of MLMOPSO in locating high quality final solutions in a lot less NOFE.

Problem 10. This problem was taken from (Runarsson and Yao2000). This problem has a very narrow feasible region comparing to the search space. The number of particles for this problem was 50 and the results shown in Figure 5.15 were obtained after 100 iteration of the algorithm with SIF factor = 2.

$Min\, f_1(x) = x^2_1+(x_2-1)^2$

$f_2(x) = x_1 - x_2$

s.t.

$h_1(x) = x_2 - x^2_1 g_2(x) = x_2 - 9x_1 \leq -1$

$-1 \leq x_1, x_2 \leq 1$

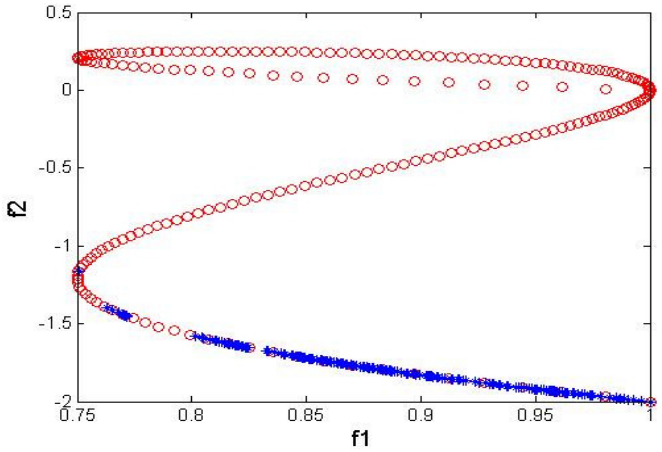


Figure 5.15- Results obtained by MOMLPSO in 100 iterations for Problem 10. Red dots show the entire feasible region and blue marks are the Pareto front obtained by the proposed algorithm

Implementation of the proposed algorithm on multi-objective problems showed the capability of this algorithm for handling non-convex constrained multi-objective problems. This algorithm is much more effective comparing to other existing versions of PSO and locates the Pareto front much more effectively in terms of number of function evaluation and also quality and diversity of the solutions.

5.15 Implication of the Results obtained on Constrained Problems

The efficient MLMOPSO algorithm was modified to handle NLP/MINP problems. The application of the modified PSO (MLMOPSO) on well-known benchmark functions in the literature showed the advantages of this algorithm in terms of efficiency comparing to the other stochastic algorithms. The number of function evaluations (NOFE) is less than all other stochastic problem algorithms which proves the efficiency of the algorithm. The algorithm also showed a very good success rate for all test functions which makes it reliable for real optimization problems. Also, using the proposed constraint handling method, all the constraints are handled simultaneously as the algorithm progresses and there is no need to deal with penalty factors or any other second optimizer such as presented by (Yiqing et al. 2007).This makes the approach

more effective, very easy to implement and independent of the constraints. It is very important in MINLP that the algorithm is capable of exploring all possible binary variable configurations while satisfying constraints. The Results obtained on the test functions demonstrate that application of the proposed constraint handling method and the applied method for handling binary variables in the novel multi-objective method provide a very powerful and time efficient optimization algorithm for MINLP and multi-objective problems. The algorithm can be easily applied for multi-objective problems which can be the subject of future studies. Although the algorithm performed well on the constrained problems due to the difficulties of handling large scale constrained problems, further exploration and improvement of the algorithm such as incorporating a method to take advantage of knowledge of the feasible region should be considered on larger constrained problems.

5.16 Self-Adaptive MLMOPSO Algorithm

There are several parameters associated with PSO based algorithms which are crucial to the efficiency and convergence of the algorithm. Nevertheless researchers have suggested different values for these parameters to improve the performance of these algorithms, the exact value of the parameters are problem dependent. There are several studies by other researchers (Chen et al., 2008; Clerc, 1999; Lu and Chen, 2008; Ratnaweera and Halgamuge 2004; Shi and Eberhart, 1999; Wu and Zhou, 2007) to make the algorithm parameters adaptive to the progress of the algorithm and dependent on the functionality of problems being handled. However, most of efforts were done on the original PSO algorithm which is not capable of handling either multiple objectives or mixed integer problems. In addition, the most of existing strategies for self-adaptive optimization algorithms are dependent on the maximum iteration number which requires priory knowledge of the problem in hand. Therefore, followings are main objectives for improving the developed algorithm.

- Make inertia and acceleration parameters self-adaptive to improve the algorithm performance dynamically while handling multi-objective MINLP problems
- Make a parameter specific to MLMOPSO algorithm, SIF, self-adaptive to improve the quality and diversity of final solution set of multi-objective problems
- Develop strategies which are less dependent on any prior knowledge of the problem in hand such as maximum iteration numbers
- Use an indicator of progress which truly measures the progress of the algorithm toward the final solution set and its closeness to convergence while it is not forcing the convergence

As mentioned in previous earlier there are several parameters associated with all PSO based algorithms which affect the performance and behavior of the algorithms. These parameters are the acceleration coefficients, c_1 and c_2 and the inertia coefficient,w, for the original PSO and all subsequent PSO algorithms and SIF, α, in addition to other parameters for MLMOPSO (the new method proposed by the authors).In PSO, inertia is responsible for balancing the capability of global and local search of the algorithm. As the inertia is increased the influence of the previous velocity on the next position of a particle is increased and the algorithm searches more globally. On the other hand, c_1 and c_2 are respectively responsible for the contribution of social (global) and personal (local) best solution in the vector of velocity. SIF factor is an additional parameter that is used in MLMOPSO method to balance the contribution of several selected leaders to leading the particles toward the Pareto front.

According to several studies (Chen et al., 2008; Clerc, 1999; Lu and Chen, 2008; Ratnaweera and Halgamuge, 2004; Shi and Eberhart, 1999; Wu and Zhou, 2007), adaptive improvement of algorithm parameters can result in locating solutions of high quality and having better convergence efficiency of the algorithm. Several concepts have been used which indicate the progress and behavior of the algorithm. Some researchers suggested a time varying adaption

strategy which is based on the number of iteration of the algorithm (Shi and Eberhart, 1999). The number of iterations indicates the progress of the algorithm toward the optimum. This is based on the assumption that as the algorithm is reaching a pre-specified maximum iteration number all particles are around the true solution/solutions. They suggested decreasing inertia using Equation 5.17. Also, Ratnaweera and Halgamuge (2004) suggested the same strategy for varying acceleration coefficients as presented in Equations 5.18 and 5.19.

$$w^t = (w_1 - w_2) * (MaxIteration - t)/Maxiteration + w_2 \quad (5.17)$$

$$c^t_1 = (c_{1f} - c_{1e}) * (MaxIteration - t)/Maxiteration + c_{1e} \quad (5.18)$$

$$c^t_2 = (c_{2f} - c_{2e}) * (MaxIteration - t)/Maxiteration + c_{2e} \quad (5.19)$$

The number of iteration is not always truly an indicator of algorithm progress toward the true solutions and does not embody the dynamic change of algorithms. Therefore, utilizing the number of iteration as the indicator of finding promising regions which encompass the optimum solutions would result in some problems such as converging to premature or local solutions. Adapting algorithm parameters when the algorithm has not found a feasible region around true global solutions would cause limitation in the global search and clustering around solution/solutions which are not optimum. Also, these strategies are dependent on a pre-specified maximum number of iterations and cannot be used when a stopping criteria is used to avoid premature convergence or extensive iterations.

Other researchers suggested strategies for improvement of parameters based on the closeness of particles to the global best in the swarm (Wu and Zhou, 2007). In these types of methods, an indicator scaled between [0 1], is defined based on the distance between the global best and the particle. This distance based indicator shows the closeness of each particle to the global best. As a particle position becomes close to the global best based on the mentioned indicator, inertia is reduced to limit the search to the local region around the global best. This approach has been used to increase or decrease the acceleration coefficients by Wu and Zhou (2007). However,

calculation of these indicators for each particle increases the computational burden of the algorithm and also decreases the global search capability of the algorithm rapidly in the case that the best particle in the swarm is not the true global best.

Lu and Chen (2008) also suggested a strategy for improving the inertia component of the updating equation based on a rough estimation of feasible region. They set both of the acceleration coefficients equal to 1 and assume that the particles are close to the global best which should be in feasible region. The magnitude of the inertia component is calculated as the deference between the particle and another randomly selected particle. The sign of the inertia component remains the same as the previous velocity. They discuss that based on the assumption that the particles are close to the feasible region, calculating the inertia component this way would limit the search to the feasible region and helps in handling constraint problems (Lu and Chen, 2008). This method would cause perplexity in the algorithm due to random change of the magnitude of the velocity and also would be misleading when the feasible region is nonlinear or disconnected and very narrow. More importantly neither of the mentioned methods is applicable for multi-objective algorithms.

5.17 Strategies for Self-Adaptive Parameters and Automated Algorithm

The main purpose of this study is to modify the algorithm parameters based on the search progress and also to minimize use of decision maker judgment in setting the critical parameters of the algorithm. The idea is that the parameters of the algorithm should be varied based on the progress of the algorithm and the shape and functionality of the problem. In this study, strategies are discussed which help the algorithm to improve its parameters based on the progress toward a promising region in the search space. Also this approach changes between constraint handling phases (using the methods mentioned in previous section) and terminates automatically. In order to develop practical strategies, two main components must be established: first, an indicator that

controls the progress of the algorithm; second, improvement strategies to modify the parameters and change constraint handling phases.

The indicator of progress control must dynamically measure the progress of the algorithm toward a true Pareto front or the global optimum. Based on the concept of PSO, it is believed that as the algorithm progresses the best particles which are known as the leaders (global bests) reach close to the feasible region which contains the global optimum solution/solutions. If there is no more change in the solutions it is assumed that the algorithm has reached the final solution. However, change in the value or positions of the solutions are not a very good indicator of the progress toward the global optimums. On the other hand, the consolidation ratio (CR) explained in section 5.8 can be used to detect changes in the *archive* which holds all the best solutions ever found in the algorithm. The CR_2 used for the purpose of parameter improvement has a lag, y, of 10 and is calculated separately. CR_2 shows the percentage of change in the members of *archive* through 10 iterations. As the algorithm more solutions close to the true Pareto optimal solutions are found which take the place of old *archive* members. The more *archive* members are close to the true Pareto optimal solutions, the less change happen in the *archive* members and they remain in the *archive* for more number of iterations. Thus CR_2 becomes larger which indicates that the algorithm is reaching to the final solution/solutions.

The algorithm should explore more of the area around the best solutions ever found to increase the quality of the solutions before convergence. Thus the parameters of the algorithm must be improved based on the progress indicator to enforce more local search around best solutions and also to increase the convergence speed and quality. This can be done by improving inertia coefficient, acceleration coefficients and also SIF factor. As it was mentioned before, inertia coefficient is responsible for adjusting between global and local search capabilities. Also acceleration coefficients are used to adjust the contribution of social and personal bests to the vector of velocity.

In the case that the algorithm is close to convergence and the improvement in the solution set is not significant, the inertia coefficient is decreased to enhance the local search capability so that the particles search more locally around the best solutions ever found. This avoids particles from hurtling too far from the promising feasible region and limits the search to that region to improve the quality of solution and locate the global solution/solutions. This also increases the convergence speed and more effective exploration of the promising area of decision making space. Reduction of the inertia coefficient, *w*, must be done gradually and based on the indicator of progress and closeness of the algorithm to the final global solution/solutions. The value of *w* must have a minimum to avoid rapid clustering and search incapability and a maximum to avoid complete random search and lack of convergence capability. The equation for gradually improving W is presented below.

$$W = W_O * exp\ (-0.6 * CR_2) \tag{5.20}$$

Where W_O is the initial value of the inertia coefficient and is set to 0.9 and the value of *W*is limited to 0.43based on Equation 5.20.

The same strategy is used for improving the acceleration coefficients. As the algorithm progresses toward the end $C_1$is decreased and C_2 is increased while their sum remains constant at a value of 4. This increases the contribution of the social component in the velocity vector and drags the particles more toward the global best solutions. The convergence efficiency is also improved due to this phenomenon. It should be mentioned that this strategy is only applied when there is more than one member in the *archive* (e.g. not while handling a single objective problem and the problem is in the second phase of constraint handling) in order to avoid clustering and premature convergence. Clustering around local or premature solutions might happen whileC_1decreases and C_2 increases significantly and permanently. However, due to the existence of the SIF factor, which is improved dynamically also, clustering does not happen when we have more than one leader (global bests). C_1 and C_2 are improved using Equations 5.21 and 5.22.

$$C_1 = C_{1o} * exp\ (-0.84 * CR_2) \quad (5.21)$$

$$C_2 = 4 - C_1 \quad (5.22)$$

C_{1o} is the initial value of the personal component of the velocity equation and is set to 2.8.

The SIF factor is dynamically increased based on the progress of the algorithm to enhance diversity in the solution set. The SIF enables use of several leaders (global bests), and thus enhances the search between leaders (global bests) and quality of solutions while handling a multi-objective problem. As inertia is decreased and the contribution of the social component of the velocity is increased, the SIF factor becomes more influential and enforces exploration between the *archive* members and the feasible area. The improvement of SIF is done dynamically since the beginning of the optimization procedure using CR_2 based on the following equation where SIF_0 is the initial value set to 0.1.

$$SIF = SIF_0 * exp\ (4.4 * CR2\text{^}0.6) \quad (5.23)$$

While the SIF is altered from the beginning of the algorithm, inertia and acceleration coefficients are improved only when the algorithm is in the second phase of constraint handling. In the first phase of the constraint handling, the goal is to locate as much as solutions that have the potential of evolving to a feasible optimum solution in the search space. Thus, the global search capability of the algorithm is strictly favorable. However, in the second phase only the most feasible solutions are present in the *archive*. Therefore, improving the parameters as described above enhances the search in the feasible region and around the best solutions that have been found in the swarm.

The proposed strategy is very effective and does not impose high computational burden on the algorithm. While this strategy provides sufficient time for the particle to more effectively enhance the quality of solutions, it does not limit the global search capability of the particles or a fraction of the swarm when the true Pareto front or global optimum has not been found.

Switching between the two phases of constraint handling method is also done automatically. As mentioned before, the main goal in the first phase of the constraint handling is to allow

particles with higher amount of violation from constrains but with lower value of objective functions to evolve. In handling optimization problems with nonlinear objectives and constraints, infeasible leaders could locate the feasible regions and lead the particles to feasible regions which contain the global optimum or Pareto solutions. Thus, the first phase of the constraint handling method should continue until the feasible region is located. If there is not a change in the feasibility of the best solutions in the *archive*, it can roughly be inferred that the feasible region has been detected and no more feasible non-dominated solutions can be found. At this point the algorithm should switch to the second phase of the algorithm to evolve only on the best and most feasible solutions using constraint dominancy concept. The feasibility of each *archive* members is measured by the total amount of its violation from the constraints, Equation 5.20. Therefore, one way to detect any change of the feasibility of the *archive* member is to track the change in variance and the mean of the amount of violation of the *archive* members. If the change in both variance and the mean of violation of the *archive* members is less than a threshold (switching threshold)the algorithm switches to the second phase of constraint handling as no more non-dominated solutions (in terms of feasibility) are being added to the *archive* .

The flow diagram of the proposed algorithm is presented in Figure 5.16. Also the contributions of this study which are explored in the next section can be summarized as following:

- Using a novel indicator for monitoring the progress of the algorithm
- Development of new strategies for dynamically adapting PSO parameters which are gradually adapted independent of priori set parameters dependent on the functionality of problems the variation speed of the adaptive parameters can be adjusted using coefficients and powers of consolidation ratio in Equations 5.20,5.21 and 5.23)
- Incorporating a combined two phase constraint handling method which uses two different approaches adaptively

- Making a novel PSO base algorithm which is capable of handling multi-objective and mixed integer nonlinear optimization problems (MLMOPSO algorithm) completely adaptive
- Improvement the performance of MLMOPSO algorithm

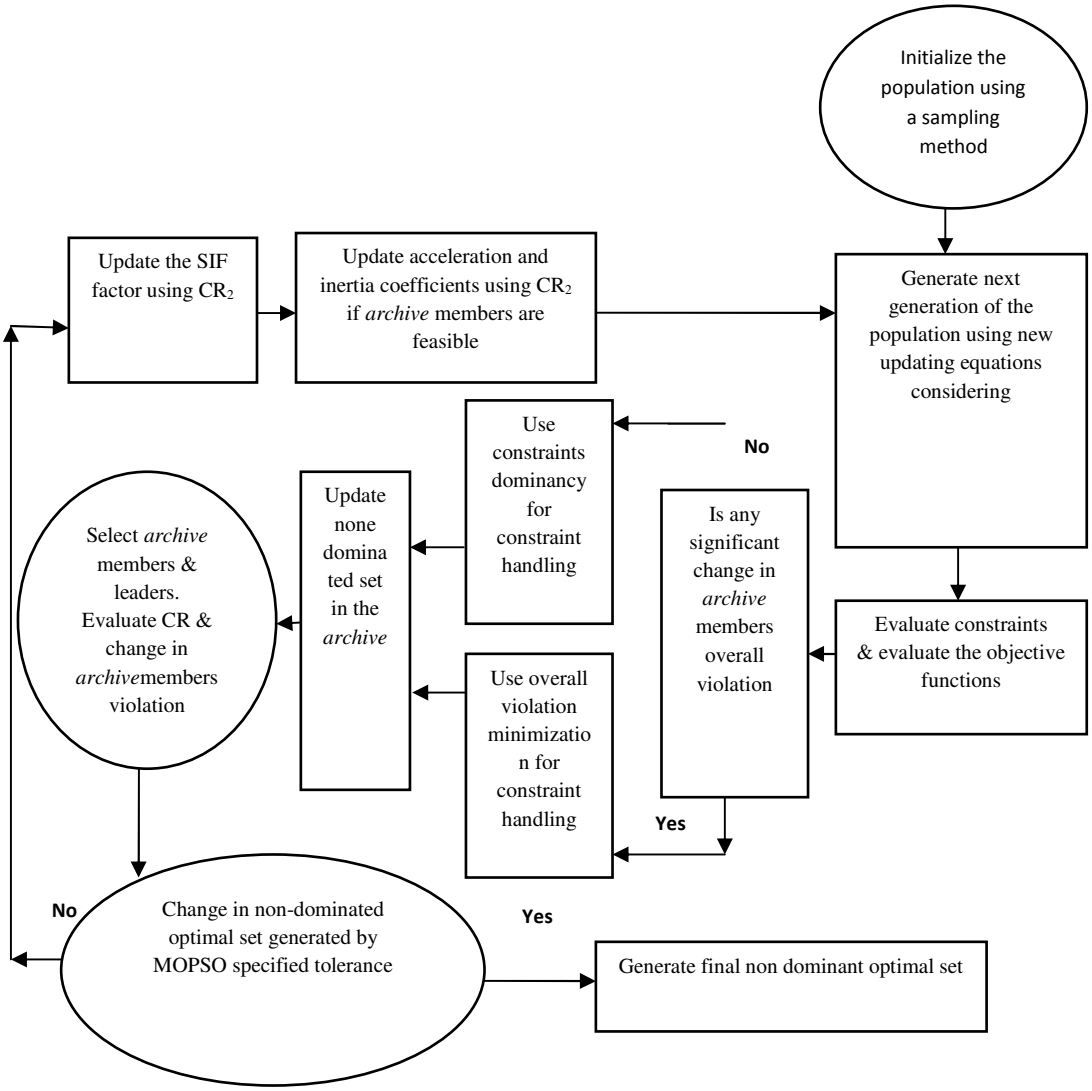


Figure 5.16: Flow diagram of self-adaptive MLMOPSO algorithm

5.18 Experimental Results and Discussion on Self-Adaptive MLMOPSO

The self-adaptive multi-objective PSO algorithm (self-adaptive MLMOPSO) proposed in this study was tested on several well-known benchmark optimization problems including single and multi-objective mixed integer problems. The results obtained from these implementations are presented and compared with algorithms proposed by other researchers. Table 5.18 summarizes the self-adaptive parameters of the MLMOPSO algorithm and their initial values and their effects on performance of the algorithm.

Table 5.18- Summary of the self-adaptive parameters and thresholds associated with MLMOPSO algorithm

Parameter	Brief description and effects	Initial values/threshold		
		Prob-lems 1,2,3, 4 & 5	Prob-lems 6,7&8	Prob-lems 9 &10
Switching threshold	Determines the minimum change of both variance and mean of the *archive* member violation in consecutive iterations in order to switch between constraint handling phases. Its value is critical to locate feasible region containing optimum solutions for nonlinear constrained problems. On the other hand if it is set to too low a value it could result in excessive unnecessary iterations.	10%	25%	5%
Stopping criteria threshold on *CR*	Determines the portion of the solution set that has remained exactly the same for *y* iterations. The higher this value is the stricter is the stopping criteria. This value does not have any effect on single objective problems.	85%	60%	85%
Stopping criteria gap *y*	It determines the number of iterations that either the value of CR (for multi-objective problems) or value of the objective function (for single objective problems) needs to remain constant in order to terminate the optimization procedure.	15	15	15
Social Influence Factor (*SIF*).	This parameter is critical for multi-objective problems for balancing the social influence of leaders (global bests) in leading particles in the swarm. It helps to maintain diversity and improves the quality of the final Pareto solution set.This	0.1		

	parameter is altered to a larger value as the algorithm is closer to convergence to increase the search in feasible region and around the optimum solutions. It also prevents clustering at the end of algorithm.	
Acceleration coefficients (C_1&$C2$)	These balance the contribution of the local best and the leaders in creating the velocity vector. They affect local and global search capability and convergence behavior. It is desirable to lessen personal contribution and increase the social component effect.	C_1=2.8 &C2=1.2
Inertia coefficient (W)	This coefficient balances between global and local search capability of the algorithm. As it is reduced the algorithm searches more locally and it is desirable when the algorithm is close to convergence. Altering this parameter triggers only when the algorithm is in the second phase of constraint handling.	0.9

Problem 1. This problem is the same as problem 6 in section 5.14. The swarm size for this problem was set to 50.The optimum solution is(x_1, x_2, x_3, x_4, x_5; f) = (-1.7177, 1.5964, 1.8261, 0.7582, 0.7690; 0.05394).

The results obtained on this problem out of 50 implementations are presented in Table 5.19 along with the result of 4 other algorithms presented in the work of Yiqing et al. (2007); improved PSO, R-PSO, required 30,000 iteration of the outer loop, stochastic ranking (SR) with 350,000 iteration numbers, adaptive segregational constraint handing evolutionary algorithm (ASCHEA) with 1,500,000 iteration numbers and simple multi-membered evolution strategy (SMES) with 240,000 iteration numbers. These results can be found in the work by (Mezura-Montes et al., 2005). Also Lu and Chen (2006) presented a self-adaptive PSO algorithm (SAVPSO) for single objective problems that the result obtained by this algorithm is also presented in Table 5.19. SAVPSO located an optimum solution for this problem in 50,000 iterations and this solution is slightly better than the actual optimum due to constraint relaxation. Again results for this nonlinear function obtained by self-adaptive MLMOPSO algorithm with a lot less number of iteration and the number of function evaluation using the proposed algorithm.

The lowest number of function evaluations (NOFE) was 9650 and the highest was 15000. Reported solution for regular MLMOPSO in section 5.14 was obtained in 10000 iterations which is in the same order of NOFE for self-adaptive MLMOPSO.

Problem 2. This problem is the same as problem 7 in section 5.14. which has three nonlinear equalities. For this problem the optimum reported in the literature is x* = (679.9453, 1026.067, 0.1188764, −0.3962336), where f(x) = 5126.498 and the overall violation for this solution is 0.00037699, from (Yiqing et al., 2007). The swarm size of this problem was 50 and the least NOFE was 15000 and the most was 23400 using adaptive MLMOPSO while NOFE for R-PSO^{b} (Yiqing et al., 2006) was 300,000 and for SAVPSO (Lu and Chen, 2006) was 50,000. MLMOPSO as reported in section 5.14 obtained results with 30,000 NOFE. Comparison between NOFE for the results obtained by self-adaptive MLMOPSO and NOFE for the regular MLMOPSO shows improvement in efficiency of the algorithm.

Problem 3. This problem was taken from (Runarsson and Yao, 2000). This problem has a very narrow feasible region comparing to the search space. The number of particles for this problem was 30 and the least NOFE for this problem was 2340 and the most was 2910.

$Min\ f_1(x) = x^2_1+(x_2-1)^2$

s.t.

$h_1(x) = x_2 - x^2_1$

$x_1 \leq -1$

$-1 \leq x_1, x_2 \leq 1$

Table 5.19 presents the results obtained for Problems 1 to 4 by the self-adaptive MLMOPSO in 50 runs of the algorithm on each problem. For all problems 1, 2 and 3 the proposed algorithm (Self-Adaptive MLMOPSO) located the optimum solution precisely and much more efficient in terms of the number of function evaluations comparing to the other algorithms.

Table 5.19- Comparison of results obtained by self-adaptive MLMOPSO and other algorithms for Problems 1, 2 and 3

Problem Number		Optimum	SR[b]	ASCHEA[b]	SEMS[b]	R-PSO[b]	SAVPSO *	Self-adaptive MLMOPSO
		Optimum/NOFE[i]						
1	Best solution	0.0539489	0.05395/ 350,000	–	0.05398/ 240,000	0.05394/ 30,000	0.0538666 / 50,000	0.053948/ 11,100
	Worst solution	0.0539489	0.21691/ 350,000	–	0.16638/ 240,000	0.05398/30,000	1.85/ 50,000	0.0541/ 12,150
2	Best solution	5126.4981	5126.597/ 350,000	5126.5/1,500,000	5126.599/ 240,000	5126.5053 / 30,000	5130.1575 / 50,000	5126.523/ 16,850
	Worst solution	5126.4981	5142.472/ 350,000	–	5304.167/ 240,000	5126.6958 / 30,000	6111.5135 / 50,000	5126.634/ 19,300
3	Best solution	0.75	NA	NA	NA	NA	0.749/ 50,000	0.74991/ 2,790
	Worst solution	0.75	NA	NA	NA	NA	0.749/ 50,000	0.749/2,340

-converged to a non-optimal solution
[i]-NOFE = number of function evaluations
[b]-LuoYiqing et al. (2006)
*Lu and Chen (2006)

Problem 4 and 5. Problem 4is the same as problem 3 in section 5.14 and has a global optimum of $(x_1, x_2, y; f) = (1.3750, 0.3747, 1; 2.1247)$. Problem is the same as problem 4 in section 5.14 with a global optimum as: $(x, y_1, y_2, v_1, v_2; f) = (13.4258, 1, 0, 3.514237, 0; 99.239641)$.

For problems 4 and 5, the size of the swarm was 50. The result of the implementations for these three problems are presented in Table 5.20 and are compared with the results presented in the work by Yiqing et al. (2007) in terms of number of function evaluation (NOFE) and percentage of run converged to the global optimum. It must be mentioned that none of the equality constraints was relaxed for testing the self-adaptive MLMOPSO in this study as opposed to other stochastic methods presented in Table 5.20. Also due to the strict stopping criteria and not relaxing the equality constraints NOFE for self-adaptive MLMOPSO is more than regular MLMOPSO for problem 4 and also more than R-PSO for problem 5. Therefore, the proposed self-adaptive algorithm was dealing with 7 continuous and 2 binary variables instead of only3 continuous variables. Table 5.20 presents average number of function evaluation (ANOFE) and percentage of run converged to the global optimum (NRC) for each problem using MLMOPSO.

Table5.20 - Results of NOFE/NRC for Problems 4 and 5 using available algorithms

Problem Number	GA[a]	M-SIMPSA	R-PSO[b]	MLMOPSO	Self-Adaptive MLMOPSO
4	22489/100	42295/100	4000/100	2500/100	4950/100
5	102778/60	63751/97	30000/100	3500/100	12500/100

[a]-Costa and Olivera (2001)
[b]-LuoYiqing et al. (2006)

Performance of the self-adaptive MLMOPSO algorithm was investigated on several well-known multi-objective test functions. The results of these implementations are presented below and are compared with either true Pareto solution obtained mathematically of with the best reported solution.

Problem 6. This problem is the same as problem 9 in section 5.14 with two objectives and two inequality constraints.

$Min\ f_1(x) = x_1$

$f_2(x) = (1+x_2)/x_1$

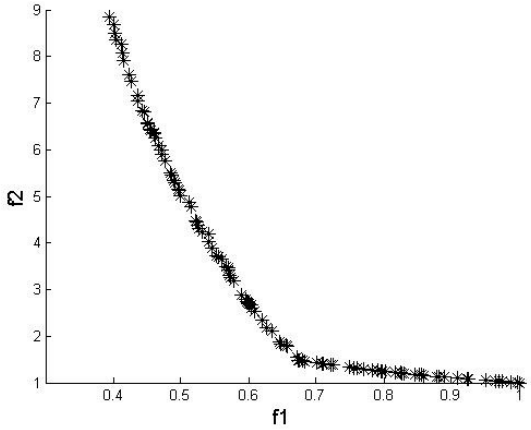
s.t.

$g_1(x) = -x_2 - 9x_1 - 0.55 \leq -6$

$g_2(x) = x_2 - 9x_1 \leq -1$

$0.1 \leq x_1 \leq 1$

$0 \leq x_2 \leq 5$



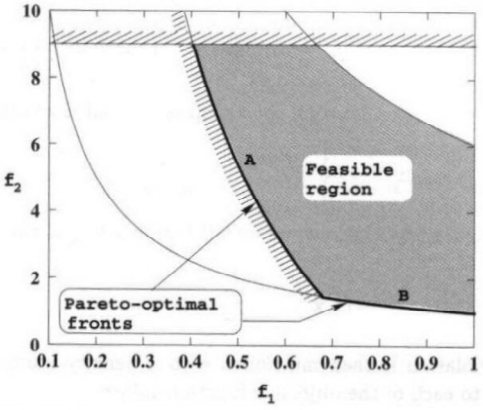


Figure 5.17- Comparison of the results obtained by MOMLPSOin48 iterations and true Pareto front of Problem 6

The number of particles for this problem was 30 and NOFE for the results shown in Figure 5.17 was1440 which comparing to the results reported in section 5.14 (NOFE of 1500)for non-adaptive MLMOPSO shows more efficiency.

Problem 7. This problem was taken from (Srinivas and Deb, 1994).

$Min\ f_1(x) = 2+(x_1-2)^2+(x_2-1)^2$

$f_2(x) = 9x_1-(x_2-1)^2$

s.t.

$g1(x) = x_1^2+x_2^2 \leq 225$

$g2(x) = x_1-3x_2+10 \leq 0$

$-20 \leq x_1, x_2 \leq 20$

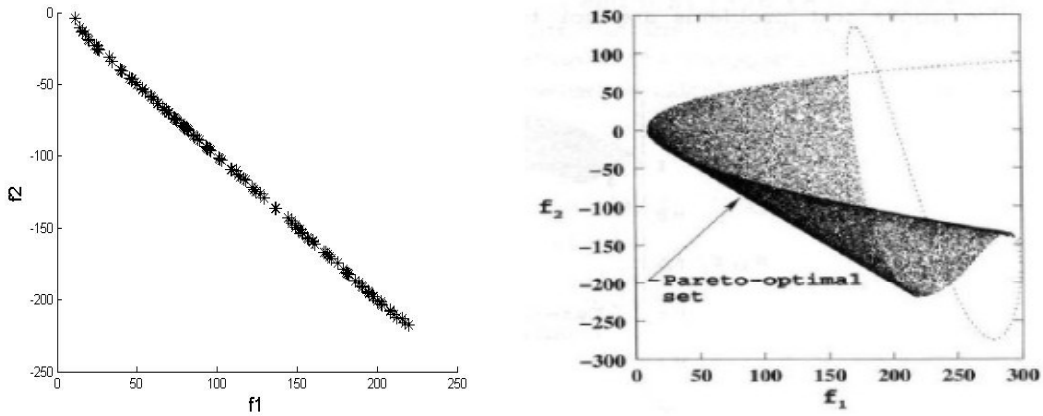


Figure 5.18- Comparison of the results obtained by MOMLPSO in 57 iterations and true Pareto front of Problem 7

The swarm size of the self-adaptive MLMOPSO algorithm for this problem was 30 and NOFE for the results shown in Figure 5.18 was 1710.

Problem 8. This problem was taken from (Binh and Korn, 1997).

$Min\ f1\ (x) = 4x_1^2+4x_2^2$

$f2\ (x) = (x_1-5)^2+(x_2-5)^2$

s.t.

$g1(x) = (x_1-5)^2+x_2^2 \leq 25$

$g2(x) = -(x_1-8)^2-(x_2+3)^2 \leq -7.7$

$0 \leq x_1 \leq 5$

$0 \leq x2 \leq 3$

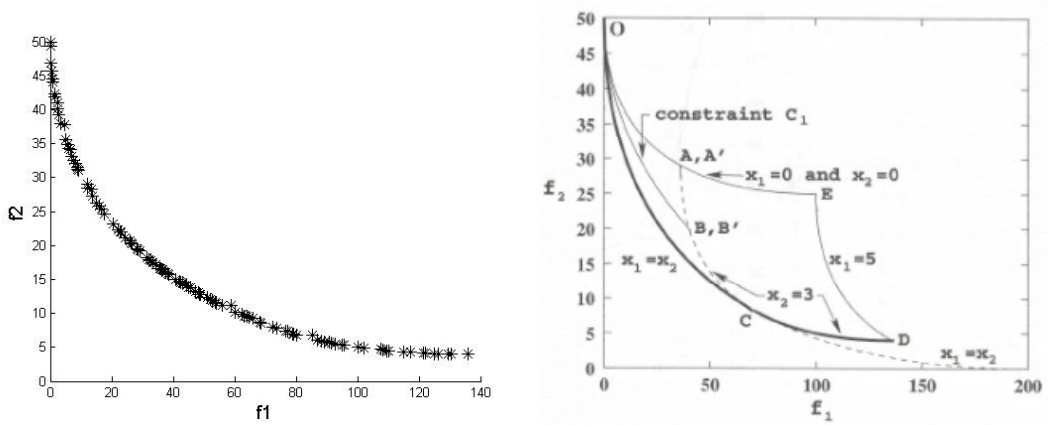


Figure 5.19- Comparison of the results obtained by MOMLPSO in 28 iterations and true Pareto front of Problem 8

Like two previous problems the swarm size for Problem 8 was 30 and NOFE for the results shown in Figure 5.19 was 840.

Problem 9. This problem is the same as problem 10 in section 5.14. This problem has a very narrow feasible region comparing to the search space. The number of particles for this problem was 40 and the results obtained with 3360 function evaluations (NOFE) are shown in Figure 5.20. Comparison of NOFE for self-adaptive MLMOPSO with NOFE reported for regular MLMOPSO in section 5.14, 5000, shows the superior performance of self-adaptive version of MLMOPSO.

$Min\ f_1(x) = x_1^2+(x_2-1)^2$

$f_2(x) = x_1 - x_2$

s.t.

$h_1(x) = x_2 - x_1^2$

$x_1 \leq -1$

$-1 \leq x_1, x_2 \leq 1$

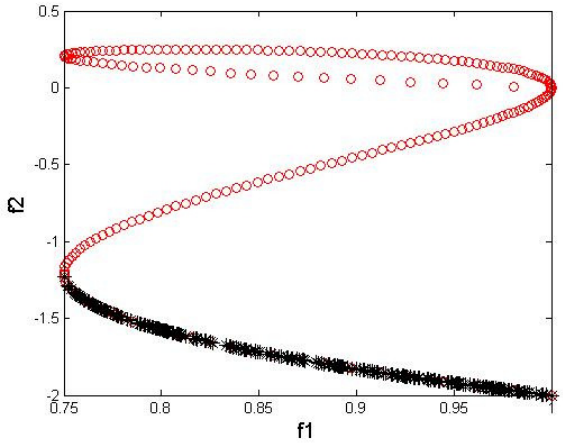


Figure 5.20- Results obtained by self-adaptive MOMLPSO with NOFE of 3360 for Problem 9. Red dots show the entire feasible region and black marks are the Pareto front obtained by the proposed algorithm

Problem 10. This is a NLP problem similar to Problem1but with two objectives. This problem was studied by Jin X. (2005) as a multi-objective problem and by Runarsson and Yao (2000) as a single objective problem.

$Min\ f_1(x) = e^{x1x2x3x4x5}$

$f_2(x) = x_1 + x_2 + x_3 + x_4 + x_5$

s.t.

$h1(x) = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 - 10 = 0$

$h_2(x) = x_2 x_3 - 5x_4 x_5 = 0$

$h_3(x) = x^3_1 + x^3_2 + 1 = 0$

$-2.3 \leq x_i \leq 2.3\ (i = 1, 2)$

$-3.2 \leq x_i \leq 3.2\ (i = 3, 4, 5)$

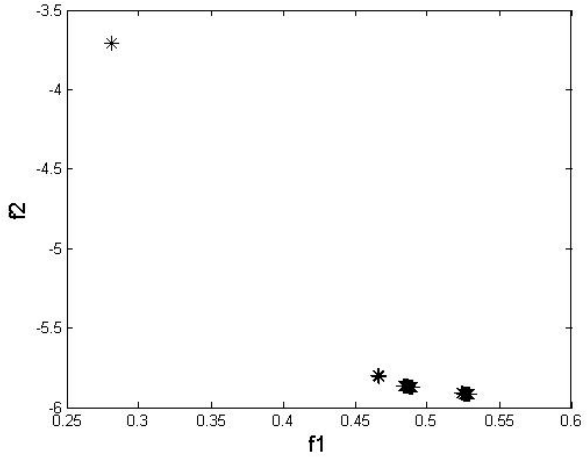


Figure 5.21- Pareto set obtained by self-adaptive MOMLPSO with NOFE of 15150 for Problem 10.

As mentioned for problem1 the optimum solution this problem with only the first objective is: (x_1, x_2, x_3, x_4, x_5; f) = (-1.7177, 1.5964, 1.8261, 0.7582, 0.7690; 0.05394) and the value of the second objective at this extreme is 3.232. Although the proposed algorithm could not locate this extreme solution while handling this problem as a multi-objective problem, it locates 209 Pareto solutions within 303 iterations with NOFE of 15150 shown in Figure 5.21. These results have not been reported yet and only two Pareto solutions were found in the work by Jin X. (2005). Also the self-adaptive MLMOPSO algorithm is still much more effective comparing to other algorithms, mentioned in Problem 1, in terms of number of function evaluation even in handling highly nonlinear multi-objective problems with several active constraints. The only two solutions reported by Jin X. (2005) are shown in Figure 5.22. However, it should be mentioned that the small red dots shown in his work do not cover the whole feasible region.

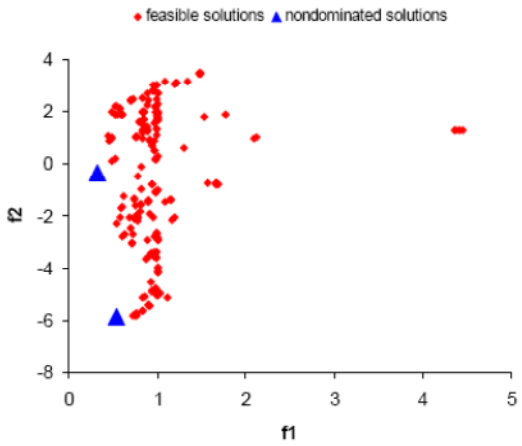


Figure 5.22- Pareto set obtained by Jin X. (2005) presented as Blue triangular points

5.19 Implications of Results Obtained by Applying Self-Adaptive MLMOPSO

Strategies for altering the parameters associated with a Multi-Leader Multi-Objective Particle Swarm Optimization (MLMOPAO) were introduced. These strategies are based on an indicator which measures changes in the solution found by the algorithm and closeness to convergence. Thus, the proposed strategies are novel from the aspect of being independent from the maximum number of iterations or any other pre-specified parameter which requires pre knowledge about the functionality of the problem in hand. These strategies help to enforce the search in feasible region and maintain diversity and quality in the final solution while increasing the rate of convergence. Also switching between constraint handling phases and termination of the algorithm were automated to decrease interference of the decision maker in the optimization procedure. On the other hand, the proposed algorithm and its parameter adaption strategies is quiet novel as they are able to handle both multi-objectives and mixed integer problems while the original PSO and its modified versions are not.

The implementation of the proposed self-adaptive algorithm on well-known benchmark optimization problems proved the capability of the algorithm in handling various types of optimization problems including single and multi-objective problems with highly non-convex and

mixed integer search space. The proposed algorithm converged to the solution precisely with a less number of function evaluations comparing to the other algorithms except for problem 4 and 5where the proposed algorithm converges with more function evaluations comparing to its original version (non-adaptive MLMOPSO) and R-PSO. This happened due to the application of strict stopping criteria which required more iteration of the algorithm based to ensure invariable solutions. This chapter introduced a new stochastic optimization algorithm capable of handling various types of problems whereas, none of the existing optimization algorithms has been reported to be well suited for handling a combination of complexities- mixed integer non-convex search area with multiple objectives- in more reasonable computational expense.

The results presented in this chapter imply that the proposed self-adaptive algorithm is applicable to a vast area of multi-criteria decision making problems dealing very complicated objectives and constraints and is not restricted to the tested benchmark functions. This broadens the application of this algorithm to engineering problems which almost always result in very complicated multi-criteria decision making problems. This algorithm enables engineers to find the optimum solutions reliably.

CHAPTER 6

MULTI-CRITERIA PROCESS DEVELOPMENT FRAMEWORK: A CASE STUDY ON SUSTAINABLE DEVELOPMENT OF DME PROCESS

The key role of multi-objective process optimization in process design can be justified through several examples such as increasing the profit of a process by with the aid of optimum flowsheet design and operating condition selection while increasing the flexibility and construability of a process and many other examples. However, one of the most regarded needs for it is justified by high demand of communities for sustainable development and specifically inherent sustainable development in chemical processes as main sinks of resources and also main contributors to negative social and environmental impacts. On the other hand multitude of conflicting sustainability criteria which are not scalarizable in most cases presents a challenging task in process design. Thus this chapter describes the proposed multi-objective process design framework through incorporating many objectives of sustainability into process design and synthesis.

6.1 Incentives for Development of a New Framework

Due to the role of synthesis in selection and integration of components of a process, it is an innovative sustainable development to incorporate sustainability criteria into process synthesis. In last decade many researchers contributed in this field to develop applicable strategies for inherent sustainable development of processes. Researchers have been used two main different category of optimization method for process synthesis: heuristic and deterministic methods as reviewed in Chapter 2.

Many approaches and methodologies have been reviewed in Chapter 2 Section 2.3.2 section which incorporated multi-objective optimization concepts into design. Although some of them have not been strictly applied for sustainable development of whole process, modified version of them can be envisioned for sustainability applications. As computational capacities has increased, process systematic design (PSE) has been facilitated and scope of application of mentioned existing methods could be enhanced to sustainable design of whole processes. However, all the mentioned methods suffer from at least one of the following issues:

- Many of proposed methodologies are inefficient in solving multi-objective problems to generate diverse and consistent Pareto optimal solution set. Deterministic methods despite of their efficiency in handling variety of MINLP problems are examples for this deficiency.
- The proposed methodologies which enable handling multiple conflicting objectives require expensive computations. Thus, handling design problems using these approaches require simplification of the process model which in turn results in less reliable solutions.
- Most of the multi-objective based methodologies do not offer a decision making procedure for helping the decision maker to select single final solution therefore, impose large cognition load on DM.

- Most of the proposed algorithms are problem specific and require significant modifications to be applied on new problems. This reduces the generality of methods.
- Despite of challenges for incorporating the conflicting objectives of sustainability in process synthesis and design, there is another key issue in this field. This problem can be regarded as how to quantitatively and generically evaluate process alternatives in terms of environmental and social impacts.

Bearing in mind the mentioned drawbacks of the existing algorithms and challenges in the field of incorporation of sustainability in process synthesis, there is a need for enhancing and developing process system engineering tools which justifies development of general frameworks and methodologies for integrating all aspects of sustainability into design (Barnicki and Siirola, 2004).

The effective novel multi-objective optimization algorithm developed in Chapter 5 and the sustainability metrics programmed as a computational tool (Chapter 3) enable development of a generic framework for rigorous process synthesis with a multitude of objectives. In this chapter, successive sections explain the modules of the proposed process synthesis framework and the concept behind them and the application of the framework is presented through a case study.

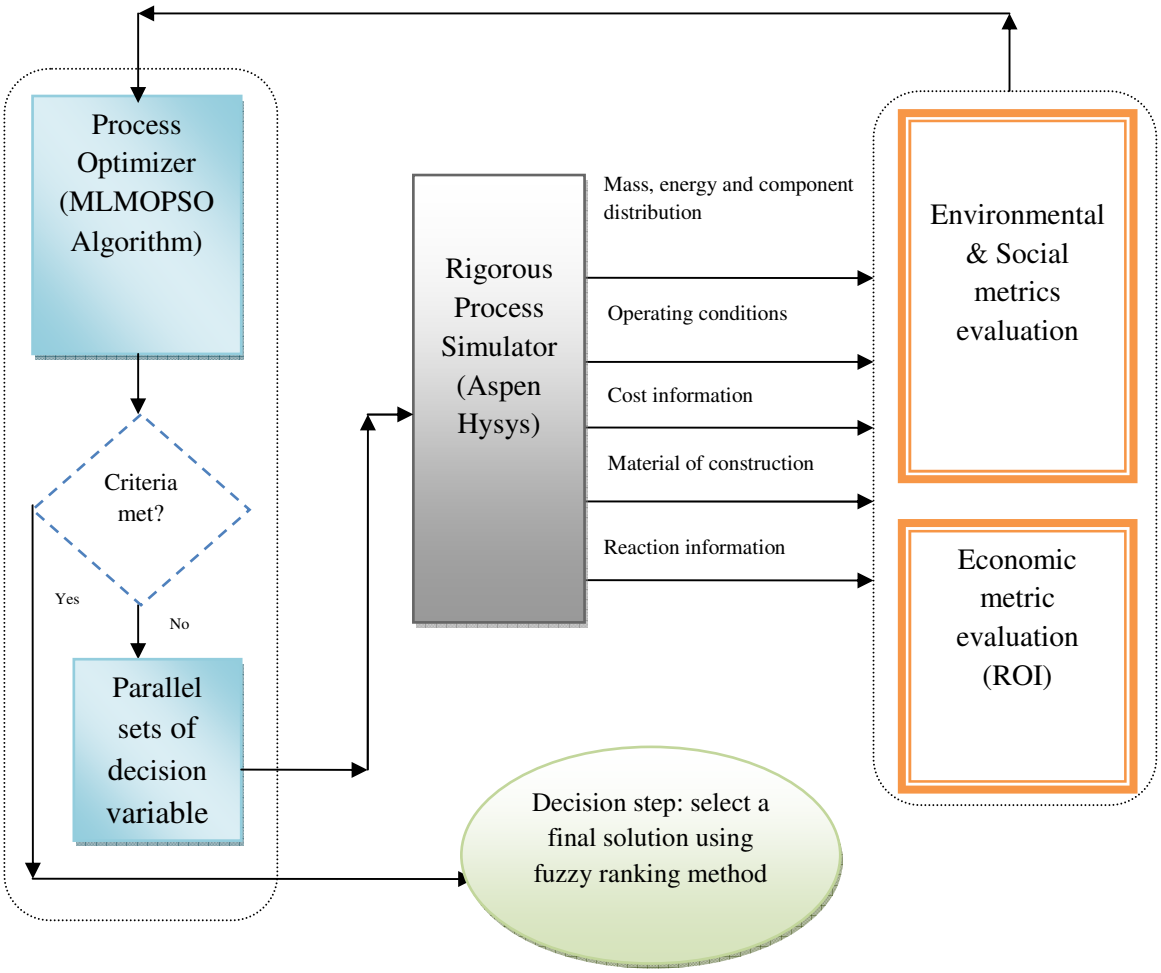
6.2 Structure and Modules of the Proposed Framework

This framework consists of four main modules: the process optimizer (MLMOPSO), a rigorous process simulator module, a sustainability evaluator and a decision making module. The overall structure and interconnection of the modules is illustrated in Figure 6.1.

The heart of this framework is the optimization method which was explained in detail in Chapter 5. The other modules, the concept behind each module and interconnections between the modules are explained in successive sections of this chapter. In order to perform a process synthesis optimization using this framework the following steps are implemented:

1. Simulation of process alternatives in the form of a superstructure in the process simulator and updating the input-output set in the simulator.
2. Determination of decision variables and defining them in the process optimizer module.
3. Updating the cost information in the cost evaluation module.
4. Run the program and insert decision making preferences to obtain a final sole optimum solution.

The computer programs for each module as well as the MATLAB code for connecting different modules are presented in Appendix B.



Figure, 6.1 the structure of the proposed framework

6.3 Superstructure Simulation in a Process Simulator

Process alternatives can be represented in the form of a process superstructure all together for the purpose of process optimization. A superstructure contains all possible connections and operations in a process. The connections of alternative streams and unit operations are represented using nodes in a superstructure. The reader should consult the book by Biegler et al. (1997) for further information on process superstructures.

In order to simulate process alternatives all the possible unit operations and streams are simulated in the simulator. Each node of the process which represents the connections of several streams is simulated using a splitter in the simulator. Also the existence of unit operations in the process is determined by the streams which are entering to that unit operation. In a particular splitter which represents a node connecting several streams, if a stream is not existed the fraction of material in the splitter for that splitter is set to zero. However, an existing stream gets a fraction value of between zero and one excluding one this means the flow ratio of an existing in the splitters representing process connection nodes would be any value greater than zero but equal or less than one. If the flow ratio of a stream is one in the splitter it means only that stream is existent in the process and the other two streams and their downstream facilities are omitted from the simulation due to no flow being sent to them. If the flow ratio is a value greater than zero but not equal to one it means only a fraction of the main stream is sent to corresponding to the value is sent to the downstream operation. Although, fractions (flow ratio) in the splitter unit determine the simulation of corresponding units and streams by the simulator, use of the binary variables are essential for formulation of the optimization problem and enforcing the physical and thermodynamic fundamentals using constraints.

Figure 6.2 represents an example of a connection node in a super structure. There are three options in the process for treating the gas stream coming separated in an oil and gas two phase separator. The sour gas can be sent directly to the flare system to be disposed or it can be sweetened by two well-known separation methods, physical adsorption and chemical absorption.

In order to illustrate the mathematical formulation, this node is formulated for the scenario that only one of the separation units can be present in the process however disposal option is necessary for some emergency situations and need to be present along with other separation units.

$$X_0 = Y_1 \times X_1 + Y_2 \times X_2 + Y_3 \times X_3 \quad (6.1)$$

$$Y_1 + Y_2 < 2 \quad (6.2)$$

$$Y_1 + Y_2 + Y_3 \geq 1 \quad (6.3)$$

While $X \in [0\ 1]$ presents the flow ratio of each stream going out of the splitter and $Y \in \{0, 1\}$ represents the existence of that stream, the first constraint only ensure mass balance around the splitter. Also flow ratios in the splitter determine the distribution of the main gas stream between process options. The second constraint ensures existence of only one of the separation units while it is possible that none of them is existent in the process. The third constraint is essential to ensure existent of at least one of the options in the process.

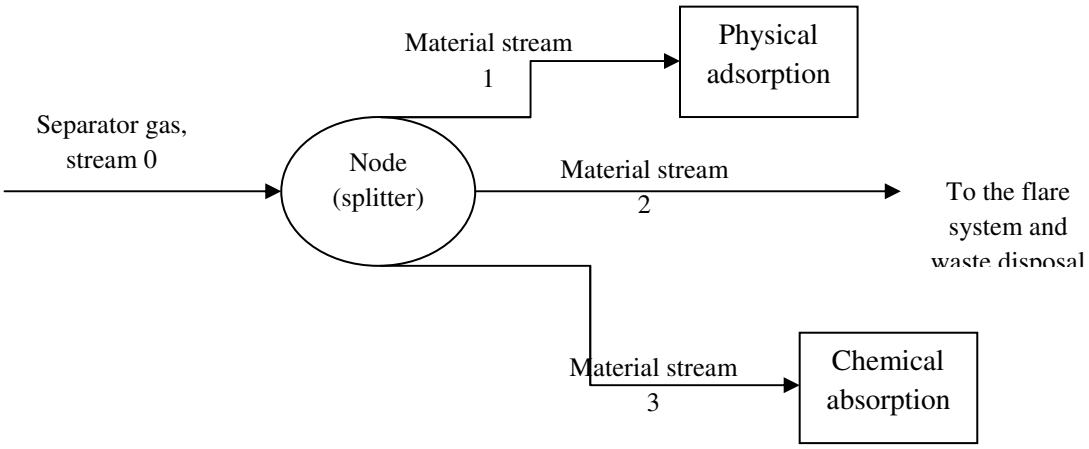


Figure 6.2, an example of a process node represented by a splitter in the process

6.4 General Design Criteria (Sustainability Module)

A sustainability evaluator module embedded in the optimization algorithm is used to quantitatively evaluate the objectives listed as sustainability concerns in the following table 6.1 to 6.3.

In this work, the sustainability metricsare programmed in MATALB as the sustainability evaluator module to communicate with the optimization algorithm and is implemented to quantify optimization objectives. The inputs into sustainability evaluator are the operating conditions, the mass and energy flows, capital costs, raw material and product costs and construction material from a process simulator and the outputs of this module are the sustainability metrics presented in Chapter 3.

The environmental impact is evaluated by using metrics developed by the Institute of Chemical Engineers (Bellof et al., 2002), GlaxoSmithKline Pharmaceuticals (Constable D., 2002) and Bridges to Sustainability (Tanzil and Beloff, 2006). This metric system has nine impact categories as listed below in Table 6.1 whose definitions and detail calculation methods are presented in Chapter 3 with details. Environmental metrics are not scalarizable and thus cannot be combined. Hence, each metric should be considered as a separate objective function.

Table 6.1-Environmental Metrics

1. Atmospheric acidification (Tanzil and Beloff, 2006)	6. Aquatic acidification (hydrogen ions equivalent)
2. Global warming (CO2 equivalent)	7. Aquatic oxygen demand (oxygen equivalent)
3. Stratospheric ozone depletion (trichlorofluoromethane equivalent)	8. Ecotoxicity to aquatic life (copper equivalent)
4. Photochemical smog formation (ethylene equivalent)	9. Resource usage (Constable, 2002; Tanzil and Beloff, 2006) (effective mass yield, e-factor, atom economy, mass intensity, mass productivity, reaction mass efficiency, material intensity, energy intensity/ fossil fuel usage and water consumption)
5. Eutrophication (phosphorous equivalent)	

Social impacts affect society as a whole. In this work, we focus on process safety risk and health index. Process safety risk is evaluated by implementing the index developed by Heikkila (1999) and using data from the International Agency for Research on Cancer (IARC) to identify the carcinogenic risk. The metrics are listed below in Table 6.2. As proposed by Heikkila (1999) a cumulative safety index which is calculated as the sum of all safety indices is used to evaluate the health impact of processes. The range of total safety score for chemical processes varies between 0 and 100 which is considered as an extremely unsafe process Shadiya (2010).

A cumulative sum of indices presented in Table 6.3 is used as the total health impact of the process. In order to calculate the indices for each chemical, its amount emitted to the environment is multiplied by its corresponding score. The list of toxic chemicals for each index can be found in Shadiya (2010).

Table 6.2-Safety metric components

1. Heat of main and side reaction Index	6. Inventory Index
2. Flammability Index	7. Temperature Index
3. Explosivity Index	8. Equipment Process Safety Index
4. Toxic Exposure Index	
5. Corrosive Index	9. Process Safety Structure Index

Table 6.3- Healthindex components

1. Carcinogenic Risk	8. Respiratory System Damage
2. Immune System Damage	9. Cardiovascular System Damage
3. Skeletal System Damage	
4. Developmental Damage	10. Endocrine System Damage Liver Damage
5. Reproductive System Damage	
6. Nervous System Damage	11. Kidney Damage
7. Sensory System Damage	

Considering some of the metrics as constraints in optimization formulation is not reasonable due to the lack of prior knowledge about them and their dependency on external factors. Also some of the objectives of sustainability may not be conflicting. However, it needs a burdensome sensitivity analysis in presence of several decision variables and may results in missing some small conflicting regions in the decision space. Therefore, to preserve the generality of the framework and explore all the possible interactions between objective functions no sensitivity analysis is performed to refine the compatible objectives. It is expected that the compatible objectives are detected by the algorithm. It should be mentioned that high dimensionality of the objective space does not affect the performance of the novel MLMOPSO algorithm thus; the framework is capable of handling a multitude of objectives generically.

The definition of some concerns is such that minimization or maximization of them is exactly the same as another concern and is not conflicting. In this work all objectives of that type will be extracted and refined to minimize the dimension of objectives.

6.5 Fuzzy Logic for Final Solution Selection (Decision Making Module)

Regardless of the category of optimization methods the final goal of a decision maker is to select the most desirable solution possible for a specific problem. As described before in Chapter 4, non-generating optimization methods cannot explore the entire search space justly and result in missing some possible optimum solutions. On the other hand, handling many objectives of sustainability with a generating method is the most generic and rigorous methodology for sustainable design of problems as it was elaborated in previous chapters. However, generating optimization methods generates many solutions which none of them are superior to others. Handling many objectives, often more than two objectives, using generating method results in many Pareto solutions thus selection of only one of them becomes very cumbersome for the decision maker. In addition, most often there is no prior knowledge about the objective functions

and their conflicting behavior with each other. Therefore, there is a need to help decision makers to select a final optimum solution conveniently.

In order to reduce the cognition load on the decision maker for selecting a final most desirable solution out of the Pareto solution set of the problem, a novel decision making module is devised for the proposed framework which interactively takes decision maker's preferences which are defined based on the objective function values obtained in the optimization step. This module incorporates the decision maker's preferences only after the optimization is completed and helps decision makers to define the preferences based on the available solutions. Therefore, there is no need for priori knowledge on the function and behavior of the problem in advance.

The fuzzy logic concept is adopted in order to define decision makers' preferences and based on the preferences an aggregated rank is assigned to each Pareto solution. The Pareto solution which has the highest rank is select as the most desirable solution.

The results in the obtained Pareto solution by the optimization algorithm represent the possible range on each objective function. The minimum of each objective function (for minimization problems) is present in the Pareto optimal solution as one of the non-dominated solutions. Therefore, the most desirable value for each objective can be presented to decision makers. The minimum value available for each objective function and maximum value of that objective in the Pareto solution set are used for defining the decision maker preferences.

After presenting the range of a particular objective function to decision makers, they asked to insert their preferred range for that objective. This preferred range is called compromised range of that objective because its lower limit may be greater than the minimum value of that objective function. Decision makers must compromise on the minimum value of some of the objectives and incorporate their preferences on the value of objective functions between the available ranges to select a single final solution. Therefore the lower limit of the compromised range for a particular objective is the value that decision makers can accept that as the compromised minimum of that objective in order to find a solution meeting their desired criteria in terms of the other objectives.

The upper the upper limit of the compromised range is the maximum value of a particular objective that can be accepted for that objective thus, any value less than the specified upper limit is acceptable and has a rank according to decision makers' preferences. After obtaining the compromised range for all objectives function from decision makers, these values are used for fuzzification of the objective functions ranges in the Pareto solution set and ranking each solution. In a minimization problem set, if the lower limit of a particular objective function entered by decision makers is less than minimum of that objective in the Pareto set it is replaced by the available minimum. The value of each objective function corresponding to a particular Pareto solution, i, is normalized as:

$$(f^{U}_{compromised,\,k} - f_{i,k}) / (f^{U}_{compromised,\,k} - f^{L}_{compromised,\,k}) \tag{6.4}$$

In order to define the rank (membership) for every solution in the Pareto solution set a strictly monotonic membership function, $\mu_{i,k}$, is used:

$$\mu_{i,k} = \begin{cases} 0, & f_{i,k} > f^{U}_{compromised,\,k} \\ D_i, & f^{L}_{compromised,\,k} \le f_{i,k} \le f^{U}_{compromised,\,k} \\ 1, & f_{i,k} < f^{L}_{compromised,\,k} \end{cases} \tag{6.5}$$

$$\mu_{i,k} = a\{\, 1 - exp\,[-b\,(f^{U}_{compromised,\,k} - f_{i,k}) / (f^{U}_{compromised,\,k} - f^{L}_{compromised,\,k})]\} \tag{6.6}$$

Where: $\mu_{i,k}$ is the membership function for solution i in terms of objective function k, $f^{U}_{compromised,\,k}$ *is* the compromised upper limit on objective function k, $f^{L}_{compromised,\,k}$ is the compromised lower limit of objective function k and $f_{i,k}$ is the value of objective function i for solution k.

However, it is also possible not to limit the upper bound of membership function to one. Cause if the objective value of a solution is less than compromised value it is more desirable. Thus, it is maybe preferable in some cases to assign a higher rank to it.

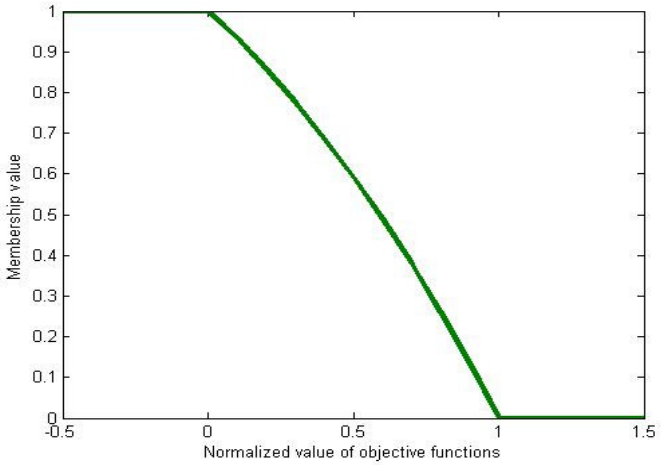


Figure 6.3- Membership function for normalized value of objective functions

Every solution in the Pareto optimal solution set is assigned an aggregated rank, $\boldsymbol{R_i}$, which represents the aggregation of membership of every objective value of it. Indeed, the aggregated rank indicates the intersection of all fuzzy memberships of a solution in terms of each objective function. Two common methods for calculation of the aggregation rank are used in the proposed framework. These two methods are presented in the text book of Sakawa(1993) and are as following:

- Convex fuzzy aggregated rank:

  $$\boldsymbol{R_i} = \sum_{k=1}^{n} \alpha_k \mu_{i,k} \qquad (6.7)$$

  Where: $\mu_{i,k}$ and α_k are respectively the membership value of solution i in terms of objective function k and weighting factor for objective k.

This method is used to find the closes solution to the preferences of decision makers. Thus, for this method the upper limit of the membership function is not limited to zero and can become negative. This helps in the case that there is no solution with a value of a particular objective function less than the upper limit of the compromised range specified by the decision maker. However, the lower limit is restricted to one to avoid obtaining larger rank for those solutions that has a lower value than the lower compromised limit for some objectives but larger value than upper compromised limit for other objectives.

- Product fuzzy aggregated rank:

$$\boldsymbol{R_i} = \prod_{k=1}^{n} \mu_{i,k} \qquad (6.8)$$

Where: $\mu_{i,k}$ is the membership value of solution i in terms of objective function k.

The product fuzzy method is used for strict selection of solution all their objective values are in the ranges specified by decision makers. If only one of the objective value of a solution is greater than the upper limit of its corresponding compromised range the fuzzy product of it becomes zero due to the limitation on the upper bound for the membership function presented by Equation 6.5.

It is expected that both methods find exactly a same solution in the cases that all objectives of at least one Pareto solutions fall in the compromised ranges. In the case that no solution is recommended by the decision making module, decision makers need to widen the compromised rage at least for some objective functions in order to find at least one solution.

6.6 Case Study

In order to describe the procedure of applying the proposed framework and test its performance on an industrial problem, Dimethyl Ether (DME) production process using ethanol as the process feed is illustrated in this section. The process synthesis framework is used to determine the best operation condition and configuration of two DME alternatives. In this section a brief description on DME base process and its modified alternative are provided. Subsequently, the procedure and methods for simulating the superstructure include both process alternatives in Aspen Hysys process simulator is presented. The next step which is selection of objective function and updating the set of process variables in optimization module is explained. Consecutively, results obtained by application of the framework are presented and compared with the results reported in other works.

6.6.1 DME Process Description

Dimethyl Ether (DME) is a propellant and also a key additive for diesel engines. DME is in the form of gas and because it is a colorless gas it is considered to be a cleaner fuel rather than other hydrocarbons. When it is combusted as a propellant or fuel additive it produces less amount of pollutants such as nitrogen oxides, carbon monoxide and it is also sulfur free. It should be mentioned that even though DME is very flammable it is not categorized as a toxicant material. The physical properties of DME are presented in Table 6.4.

Due the clean burning nature of DME it has been suggested by several researchers as a good substitution for diesel fuels, petrol engines and gas turbines (Horstman et al., 2005; Semelsberger et al., 2006; Arcoumanis et al., 2008; Savadkouhi et al., 2010). It has a vast area of appliance as the main fuel such as transportation, power generation, cooking heating etc. (Ogawa et al., 2004).

Table 6.4-Physical Properties of Dimethyl Ether

Property	Value
Boiling Point (˚C)	-23.6
Freezing Point(˚C)	-141.5
Solubility in water, 20˚C, g/L	71
Liquid Density, g/L	1.97
Molar Mass, g/mol	46.07

In some countries due to lack of other fuels DME is used as the main heating source and fuel for transportations for example in China it has been used as substitution for coal (Ogawa et al., 2004; Han et al., 2009). DME is produced by two main reaction pathways in industries: via dehydration of methanol and via natural gas. In this study the process of production of DME via dehydration of methanol is investigated.

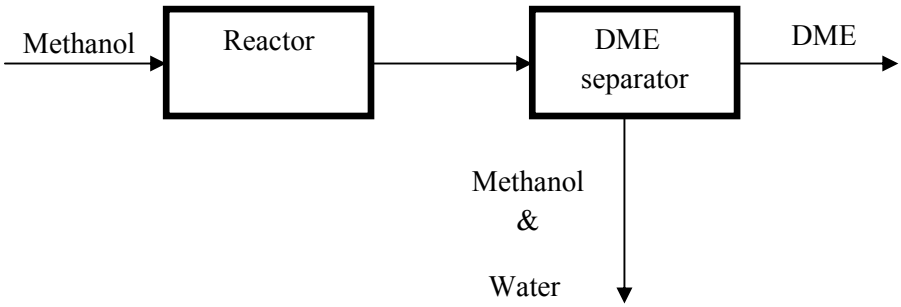


Figure 6.4, Block diagram for the base DME production process

A block diagram for the base method of DME production using methanol as the feed is illustrated in Figure 6.4. In this process, a catalyst reactor carries out the reaction which is shown by Equation 6.1 presented by (Turton et al., 2009).

$$2CH_3OH \longrightarrow (CH_3)_2O + H_2O$$

Methano DME Water (6.9)

In this process methanol is fed to the reactor with purity of 99.5% at 25 ˚C and pressure of 1 bar. The state of raw material with the mention condition is liquid so it can be pumped to the reactor which operates at 25 bar. Two shell and tube heat exchangers in series raise the feed temperature respectively to 150˚C and 190˚C which is the operating temperature of the reactor for the base case. Reactors in industries are design such that 80 % conversion of methanol is obtained in the mentioned operating condition based on reaction presented by equation 6.9. The reactor effluent is hot and it is cooled down using two sequential shell and tube heat exchangers respectively to 278˚C and 100˚C to be fed to the distillation tower. Feed is entered in distillation tower with the pressure of 10.4 bar. A pressure throttling valve located after two heat exchangers is used for reducing the pressure of reactor outlet stream. The depressurized stream which is in liquid phase is sent to the distillation tower for separation of DME from the solution. Feed enters at middle of tower which contains 20 ideal equilibrium trays without accounting for the partial

condenser and reboiler. DME exits the condenser in gas phase with a purity of 98% and is sent for further cooling down and storage.

The operating condition described above is applied in industry and was reported by Shadiya (2010). This is considered as the base case of the process and the optimized processes will be compared by the base case. It should be mentioned that the base case here has been reported by Shadiya (2010) as the most sustainable alternative in terms of all sustainability aspects between two pathways of DME production mentioned earlier. Thus, further optimization of this process regarding sustainability concerns would make this process even more sustainable to be used in industries.

6.6.2 An Alternative for DME Process Modification

Using hierarchical approach presented by Douglas (1988), the separation train of the DME process could be modified by adding a separation unit after the first distillation column for separating the unconverted portion of methanol and recycling back to the reactor to reduce the resource usage of the process. This task could be best done by adding a distillation tower for separating methanol and water. Distillation tower is the best choice for this separation task based on previous practices and thermodynamic insights based on relative volatility (Douglas, 1988). However, adding such a unit to the base process must be assessed based on the sustainability concerns. The existence of the second distillation tower for recovering methanol as the raw material must be decided based on other metrics of sustainability. Although he second distillation tower will recover the unconverted portion of raw material and will improve the resource usage and some environmental issues of sustainability, it imposes more energy requirement to the process for the separation task. Also the recycle stream also should be considered from utility consumption aspect because it would need some cooling/heating and pressurizing facility to prepare to be mixed with the fresh feed and then be sent to the reactor.

On the other hand the fraction of material to be separated in the second distillation tower could be considered as a decision variable in the design. It may not be the most sustainable option to recover all bottom product of the first tower which is disposed and sent to a waste treating unit. The block diagram for the DME with raw material recovery is illustrated in the Figure 6.5.

The second tower is designed with a partial condenser. Even though, the feed is pumped to the reactor and need to be in liquid phase, due to the small fraction of recycled stream comparing to the fresh feed the recycled vapor will be condensed when it is mixed with the fresh feed. It must be mentioned that due to sufficient high pressure of overhead product of the second distillation tower there is no need for compression which reduces high operating and maintenance cost of a compressor. Also, this would reduce the duty of tower condenser and the heat exchangers on the reactor feed stream which further justifies use of a partial condenser. In addition, control of pressure for the second column would be easier which itself could be seen as more flexibility in the process and thus more sustainability of the process.

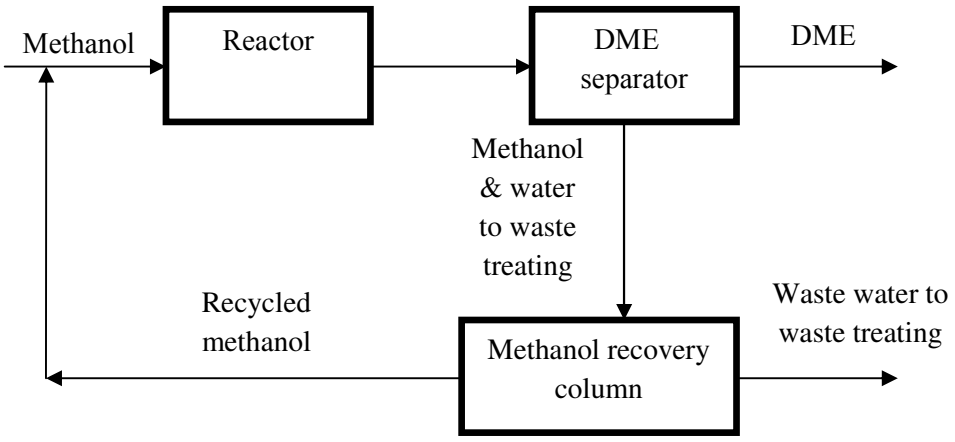


Figure 6.5, Block diagram for the modified case of DME production

The specs of the second column which are methanol recovery and purity are fixed parameters. It does not make sense to send a fraction of water and methanol mixture to the second distillation tower to recover less than 95% of it. Also the purity of recycled methanol should be consistent with the purity of the fresh feed in order not to effect the catalytic reaction in the

reactor. The other important reason for maintaining the purity of recycled methanol higher than 95% is to avoid circulating high amount of water in the system. Large amount of water being re-circulated in the system would cause several problems as:

- High pumping cost for the recycle stream.
- Larger reactor which impose much more operating and capital cost to the process.
- Larger heat duty needed for four heat exchangers on the feed and effluent streams of the reactor which increase operating cost significantly.
- And finally, more heat duty and boil up ratio of the first and second column.

A sensitivity analysis was performed for determining the least possible purity of the second column overhead so as to keep the reactor feed concentration constant. This was done while 100% of the first water bottom product was sent to the recovery unit cause this scenario corresponds to the worst case that the purity of recycled methanol could affect the reactor feed purity. The analysis showed that methanol should be recovered in the second column with purity of 95% which is also in accordance with recommended engineering practices Douglas (1988).

The decision making about the mentioned tasks above (modification of the base DME process by adding a raw material recovery unit) are subjects of process synthesis in order to find most sustainable configuration and operating condition. Relying on heuristic method and recommended practices for making decision on these design tasks may not result in the most promising design in terms of sustainability. This would happen due to that recommended practices only consider limited aspects sustainability as the design criteria and even with one design criteria it is not possible to investigate all possible options without a reliable optimization tool.

It is good to be noted that many other design parameters such as the purity of the second column overhead, mentioned earlier, could be considered as a decision variable for this process synthesis task to obtain more integrated optimum solutions and to prevent use of limited

knowledge obtained using sensitivity analysis which is always restricted to specific scenarios. However, for the sake of testing this framework with faire comparison to the other reported work on this process, this parameter and some other possible decision variables were not considered for optimization and were determined based on industrial data.

The proposed framework is applied to make decision on whether adding the second distillation column would be a good option based on sustainability metrics and decision maker's preferences. Other parameters also are considered for modification of the process as this an integrated unit which is explained in the subsequent section.

6.6.3 Simulation of DME Superstructure

As discussed in earlier sections in order to explore all possible configuration and operating condition of process using an optimization method, the alternatives of the process need to be presented in the form of a superstructure. Superstructure includes all possible unit operations and possible energy and material streams connected between process units. Existence of a particular unit operation or a stream is determined by binary variables in the optimization formulation. When the value of a binary variable (also none as class variables) get the value of one it represents a particular stream or unit operation to be existed in the process. Existence of combination of unit operations and streams or restrictions on the combination of process components are forced using equality and inequality constraints to ensure physical and thermodynamic feasibility. The reader is referred to Section 6.2 for more information on using binary variables and constraints for formulating a process superstructure.

The two alternatives of DME process (base case with and without recovery unit) are simulated in Aspen Hysys in the form of a superstructure. NRTL_RK is chosen as the fluid package (thermodynamic model) in this work because of property prediction capability of NRTL model for highly non ideal liquid solutions (Merzougui et al., 2011) and RK equation of state for vapor phases.

All the possible configurations and unit operations must be included in the simulation. Therefore, using the approach explained in Section 6.2, splitters are used to represent nodes of the superstructure and the fractions of stream (flow ratios) in splitters along with binary variables are used for determining the existence of unit operations and streams.

The connections of two process alternatives to be included in the process simulation of DME superstructure are presented in Figure 6.6.

The flow ratios of splitting node in the above superstructure determine the existence of methanol recovery system which includes the second separation tower and recycling facilities and also the water waste stream. It should be mentioned that both recovery system and methanol-water mixture waste stream coming from the bottom of the first column can be existent simultaneously. In this case the waste generated by the base case is distributed between two options which are raw material recovery and no material recovery.

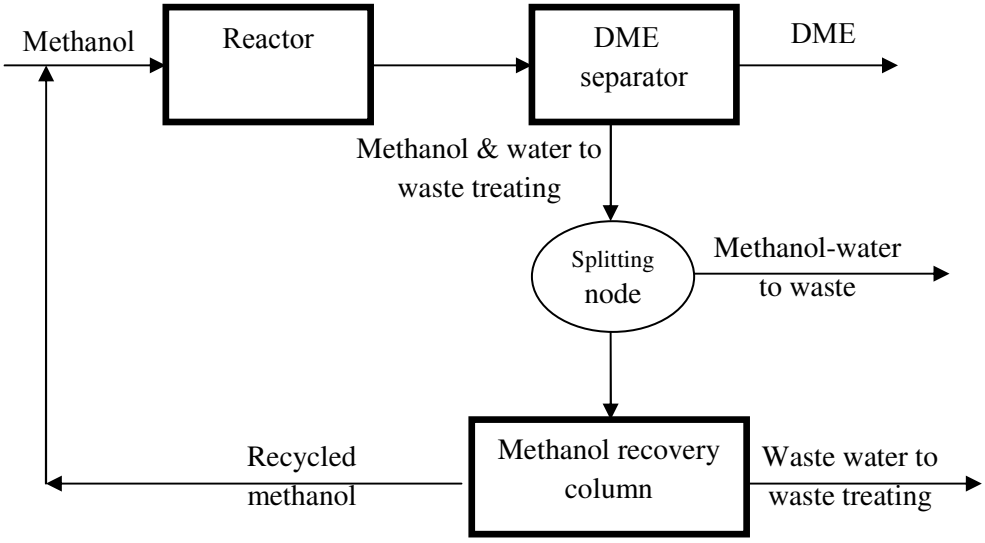


Figure 6.6, Block diagram for the two alternatives superstructure

Two cases of process synthesis are explored in this chapter:

1. DME process retrofits (modification) in which the improvement of the process is performed by adding the recovery system and varying the rector temperature and first tower pressure. It is considered as a process retrofit because no change in the existing base case unit operation is done. This also makes it easier to compare the performance of the algorithm with reported data by Shadiya (2010).
2. The second case includes more continuous decision variable including the reactor volume. As the reactor is designed and the conversion of the raw material is changed this case is regarded as a design. Also the fresh feed flow rate is another decision variable in this case while the production amount of DME is restricted due to market demands. Since this case is completely different from the industrial design and includes more decision variable, the result obtained it is presented separately.

Regardless of the retrofit or design case the overall structure of the optimization problem is the same. Based on the description provided in this chapter on formulation of a superstructure there are two discrete (binary) variables associated with this process at splitting node. The superstructure simulation in Hysys is illustrated in Figure 6.7. And the formulation for DME superstructure can be presented as following:

Minimize F: $\{f_1, f_2, \ldots, f_n\}$

DV: X, Y

Subject to:

Physical constraints:

$$r_1 \geq 0.2y_1 \tag{6.10}$$

$$r_1 \leq y_1 \tag{6.11}$$

$$y_1+y_2 \geq 1 \tag{6.12}$$

$$r_2 \geq 0.05y_2 \tag{6.13}$$

$$r_2 \leq y_2 \tag{6. 14}$$

$$y_1 \times r_1 + y_2 \times r_2 = 1 \quad (6.15)$$

Process constraints (for design case only):

$$Prod_{DME} \geq 5500 \text{ kg/hr} \quad (6.16)$$

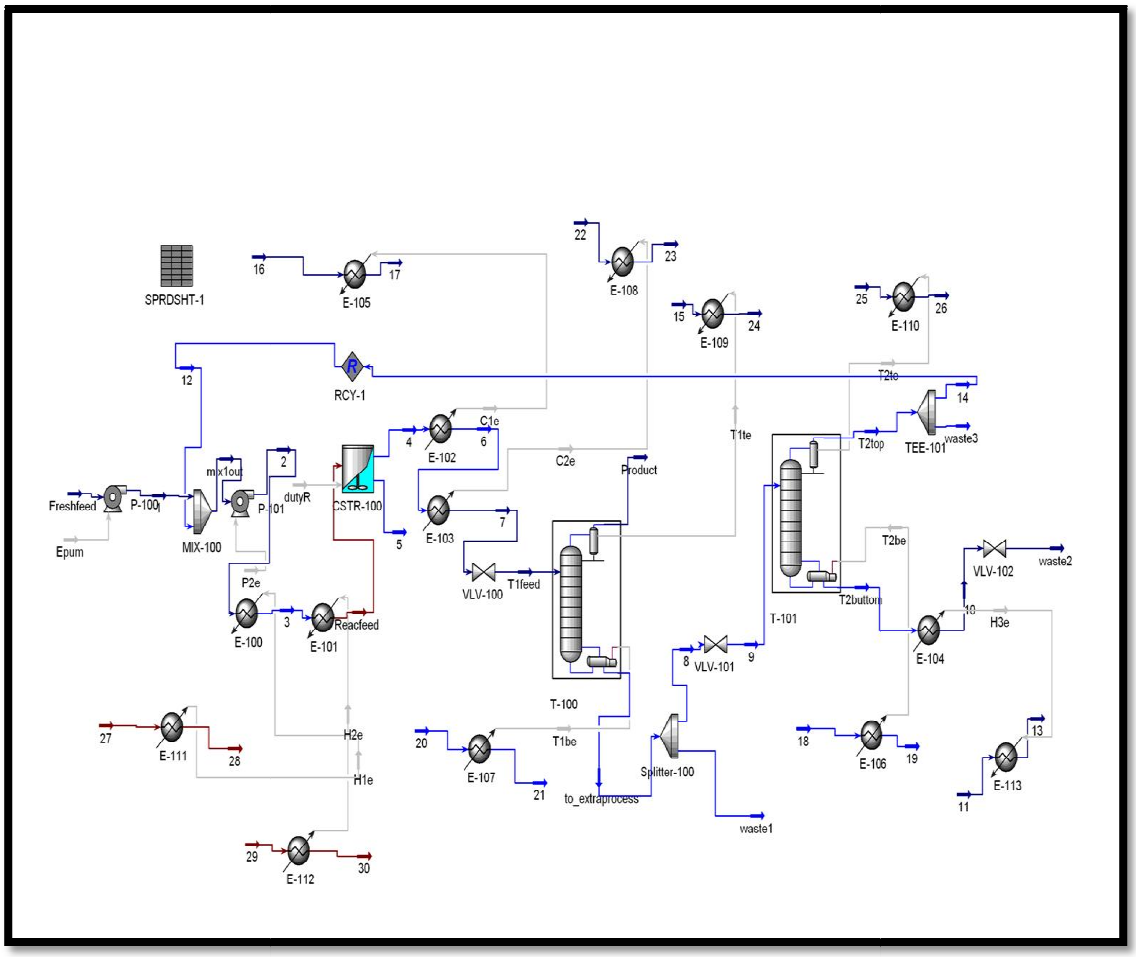
$$Prod_{DME} \leq 7000 \text{ kg/hrhr} \quad (6.17)$$

Where: r_i is the flow ratios in the splitter100 and y_i is the binary variables associated with that node. X and Yare the vector of continuous and binary decision variables respectively.

In this formulation there is no need to include process mass and energy balance constraints because those constraints are solved internally by the black box simulator (Aspen Hysys) which results in rigorous and effective design and optimization. Objectives for design of both cases are the same and along with the decision variables of each case will be discussed in subsequent sections. First and second constraints on the flow ratio of stream 8 entering the second tower are to limit its valueless than one and greater than 20% of the stream entering to the splitter. The reason for having a lower limit is to avoid zero flow in the case that the recovery unit is existent and also distillation separation would not be reasonable for a flow rate less than 20% of splitter feed. The same key roles are played by fourth and fifth constraints for the flow ratio of the stream conveying the methanol and water to the waste treatment facility. The third constraint is essential to ensure existence of at least one of the options and the last constraint enforces the mass balance around the splitter. However, the process constraints which are not necessary for physical and thermodynamic feasibility of the process are applied only for the design case to meet the demand limitations for DME production amount.

To avoid too large a number of iteration which may happen in Aspen Hysys due to lack of convergence of internal equations the simulation maximum number of iterations must be limited to a specific number which for this study was 500. To avoid fatal errors in the software due to violation from crucial physical constraints a pre evaluation module in the optimizer first evaluate the physical constraints and only if all this type constraints are met the decision variables are sent

to the next level for process simulation by Aspen Hysys. This would make the framework more timely efficient. Also in order to meet the specification of a particular process the combination of specs for iterative operation units in Hysys are selected such as to reduce the likelihood of divergence of the internal mathematical model of the simulator. For example instead of fixing the reflux ratio in the distillation columns of DME process to meet the required specifications, another spec like overhead recovery of methanol can be selected to minimize the convergence issues in the simulator. This simple trick significantly reduces divergence rate in columns and in most cases solves such problems.



However, any trial solution generated by the optimizer which causes unpredicted error and warnings from the simulator or violation from physical constraints detected in pre evaluation module is treated by big penalty factor. This way all the objective functions evaluated for the fatally violent trial solution are penalized by a sufficient big number. In handling sustainability some of objective functions may have huge value so attention must be paid in selection of big penalty values. To differentiate between fatal errors which result in complete lack of convergence and warnings which may result in some inconsistency but nor divergence the penalty value for penalization of these two types of errors are different.

6.6.4 Selection of DME Design Criteria (Optimization Objectives)

The sustainability metrics which are used for DME process are explained here. For the economic aspect of the process Return on Investment (ROI %) explained as one of the economic indices is chosen for this work which calculation of it for this process is presented in the next section. For the environmental aspect of sustainability based on the material presented in the process and list of material contributing to each metric, presented in chapter 3 in detail, only two of the environmental metrics are of the concern which are:Global warming (CO2 equivalent) and Photochemical smog formation (ethylene equivalent). Therefore, two objective functions in the framework address the environmental concerns except the resource usage.

Resource usage metrics explained in Chapter 3 are considered as objectives of this study as well as other environmental metric mentioned above. However, some of them are omitted from the set of optimization problem objectives because they become exactly the same as some other presented objective as is explained bellow.

1. E-Factor to be minimized: this is considered as an objective for the process optimization. It is the ratio of total waste over mass of product.

2. Mass intensity to be minimized: this is calculated as the ratio of total mass used in the process step over mass of the products. Total mass used in DME process is considered equal to the raw material mass due to not existence of any type of solvent of consumable material in the process and make up for utility material is ignored in this study.
3. Reaction mass efficiency to be maximized: this metric of sustainability is calculated as mass of product over mass of reactants which is the reciprocal of mass intensity metric. Therefore, minimization of the mass intensity is exactly the same as maximizing Reaction mass efficiency. Thus, this metric is omitted from the list of objective functions.
4. Mass productivity to be maximized: this metric is the ratio of mass of the products over total mass used in the process step which is again the reciprocal of mass intensity metric. Therefore, minimization of the mass intensity is exactly the same as maximizing Reaction mass efficiency. Thus, this metric is omitted from the list of objective functions
5. Effective mass yield to be maximized: this metric is calculated as the mass ratio of the desired product over non-benign reactant. In this particular case study none benign reactant is the only reactant which is methanol thus this metric becomes exactly same as mass productivity and is not considered as a separate objective.
6. Water intensity: this metric is the ratio of the consumed water over the total mass of desired product. Since this process does not consume water and produce some waste water this metric is not applicable for this process and its value is constant equal to zero.
7. Energy usage: this metric represent the consumption of energy in the process and is calculated as the ratio of net consumed energy over the total mass of desired product. This metric is considered as one of the objectives in for the optimization.
8. Atom economy: this metric is calculated as the ratio of the molecular weight of desired product over the molecular weight of reactants. As only one reaction pathway exists in this case study therefore, the value of this metric remains constant and cannot be optimized.

9. Material intensity: this index is to be minimized. However, based on the calculations presented in Chapter 3 for this index, for the DME process case this index becomes equal to mass intensity index minus one. Thus, minimization of it is exactly same as minimization of mass intensity.

Therefore, the environmental aspect of sustainability imposes four objectives for the process synthesis of DME case study.

Social impact of processes as explained in Section 6.4 is represented in this work by safety and health indices. Most of the components of the safety index are constant. Due to the fact that there is only one reaction pathway the Heat of main and side reaction index is constant. The material of construction is selected based on the service and range of pressure and temperature in the process based on available engineering standards thus the corrosion index would remain constant. Also flammability, toxic exposure and explosivity indices are constant due to the only one option for material being involved in the process. Also based on the range of optimization decision variables corresponding to temperature and pressure in the process and also the inventory amount of DME process according to the tables presented in Chapter 3 for calculation of safety indices the pressure is constant. The maximum pressure in the process is the reactor pressure which is 25 bar and is constant therefore, the pressure index is constant. However, due to changes in the maximum temperature of the process as decision variables vary, the overall safety index is variable. Thus, safety is considered in the list of objective functions. The overall health index is also varying as the configuration and operating conditions of the process change due to the change in material emissions. Therefore, the overall health index is also another objective for the process optimization of DME.

According to above explanations, the optimization of DME process has 8 objectives to be minimized. All then objective functions are programmed in the optimized module and the

computer program code is presented in Appendix B. In the subsequent sections two cases for process optimization of DME process are presented: DME process retrofit and DME process design. The decision variables associated with each case are presented accordingly.

6.6.5 Cost Evaluation of the Superstructure

Return on Investment (ROI %) index is utilized in this study for evaluating the economic aspect of DME process alternatives. ROI is calculated as:

$$ROI = 100 \times Annual\ profit / total\ investment \tag{6.17}$$

It should be noted that ROI calculated in this study is based on annual profit after tax with the rate of tax on income equal to 48%. Total investment and annual profit of the process are calculated based on the methods presented by Douglas (1988) as following:

$$Profit\ before\ tax = 0.974(revenue) - 1.03(raw\ material\ cost + utility\ cost) - 0.186(onsite) - 2.13\times10^{5} \tag{6.18}$$

Onsite cost, based on Douglas (1988) definition, is the cost of purchasing and installing all the equipment inside the battery of the process including their spares, instrumentation and all piping associated with the equipment. The onsite cost is equal to the capital cost in this study because the equipment cost estimation equations include the other costs associated with the equipment and are the installed cost of equipment. The capital cost is depreciated over the lifetime of equipment which is considered 15 years in this study (Douglas, 1988).

$$Annualized\ onsite\ cost = 0.181(Capital\ cost) \tag{6.19}$$

Raw material cost and revenue are calculated respectively by multiplication of raw material and products prices with their annual mass amount. Utility cost is also calculated using the cost of

cooling water, steam and fuel used annually in the process. The cost information for DME process is presented in Table 6.5.

Total investment cost is also calculated using the equation presented in (Douglas, 1988) as following:

Total investment = fixed cost + work capital cost + startup cost = fixed cost + 0.15(total investment cost) + 0.1(fixed capital cost) (6.20)

Work capital cost is funds needs to be available to operate the process initially. For example purchasing raw material and filling up the tanks and etc. the simplified form of Equation 6.21 is:

Total Investment = 2.36(Capital cost) (6.21)

The installed cost of the equipment presented in the process is calculated in cost evaluating module of the framework based on cost estimation models presented by Guthrie (1996). The cost estimating correlations are updated using M&S indices using the methods presented by Douglas (1988). However, it is recommended to use more update correlations for equipment cost estimation based on the data available in companies or provided by vendor for more accurate estimations. The cost correlations can be easily updated in the cost evaluation module of the framework. The computer program code for the cost evaluation module is presented in Appendix B.

Table 6.5-Summary of Economic data for the Dimethyl Ether Process

Cost ($)	Item
$0.294/kg (Turton et al., 2009)	Methanol
$0.06/kilowatt-hour (Turton et al., 2009)	Electricity
$1.17/kg (Turton et al., 2009)	DME
$14.8/1000 m^3 (Turton et al., 2009)	Colling Water
29.97/1000 kg(Turton et al., 2009)	High pressure steam

29.29/1000 kg(Turton et al., 2009)	Low pressure steam
$0.036/kg (Turton et al., 2009)	Waste Treatment

6.6.6 Optimization of DME Process

In the next two sections two different cases for optimization of DME process, design and retrofit, are explained and the obtained results using the proposed framework are presented and discussed. For the presented results in this chapter, significan figures are kept to show the capability of the proposed algorithm in locating precise solutions with very small differences although too many significan figures should be ignored in engineering applications due to existing uncertatnies.

6.6.6.1 Results of DME Process Retrofit

The base Dimethyl Ether (DME) process is retrofitted for recovering unconverted raw material and use it for the feed of the reactor again. The idea of recovering material is to reduce the waste of raw material in the process and produce more desired product to increase the profit of the process. However, there is no guarantee that adding the recovery unit to the process satisfies all the sustainability concerns and it is not an easy task without applying optimization to determine the operating conditions of the reactor and distribution of material between two process alternatives. Therefore, the proposed framework is applied for determining the best configuration of the process and its operating conditions.

In process retrofit of DME the assumption is to avoid any change in the existing process which is the base case DME process. This means that no change and modification is perform on the existing equipment such as DME reactor and the first distillation tower. Only a second distillation tower and its associated equipment and recycling piping are added in order to recover methanol and send it to the reactor inlet. Therefore, the size of the reactor is maintained constant

as the base case and is not considered as a decision variable. This implies that the conversion of the reactor is not a decision variable either but it varies slightly due to the variations in the reactor inlet flow rate which is caused by slight changes in the recycle flow rate and also variation of the reactor temperature. However, because the reactor is equipped with a heating jacket the heat duty of the heater can be varied by controlling the utility supply to maintain different operating temperature in the reactor. The operating temperature of the reactor is considered as a decision variable as it has direct effect on the conversion of methanol. In addition, operating pressure of the first distillation tower which has direct effect on separation of DME can be varied easily with changing the set points of the pressure control system of the tower with no need for changing the existing equipment.

The existence of two process alternatives embedded in the superstructure imposes other two decision variables which determine the fraction of stream going out from the first tower reboiler which should be sent to the recovery unit. The flow ratios of streams Waste1 and steam 1 shown in Figure 6.7 in the splitter is determined by two decision variable r_1 and r_2 respectively. According to the superstructure formulation also there is two binary decision variables, y_1 and y_2, which determine the existence of methanol-water waste stream going to waste treatment facility and recovery unit. The optimization decision variables and their upper and lower limits are presented in Table 6.6.

Table 6.6-Range of decision variables

Decision Variable (DV)	Range
Reactor temperature (˚C)	195-400
Distillation tower pressure(bar)	10.50 to 14
Flow ratio of stream going to recovery	0.1 to 1
Flow ratio of stream going to waste treating facility (stream waste1)	0.2 to 1

In order to determine whether adding a recovery unit and recycling system improves sustainability of DME production process, as it was described in previous sections the superstructure of the process was simulated in Aspen Hysys and the simulator was linked to the optimization algorithm and sustainability evaluator module to dynamically send the process mass, energy, operating conditions and the information required for equipment sizing to the evaluator and receive the updated decision variables from the optimizer. Sustainability evaluator in turn sends the objective values evaluated based on the information from the simulator to the optimizer. The framework was run and the Pareto solution set for this problem was found in within 50 iterations of the optimization algorithm. The swarm size in the optimization algorithm was 50. Therefore, the number of function evaluation for this problem was 2500 which is considered a low number and proves the efficiency of the algorithm. However, the total run time for this problem was 25 minutes which is a reasonable time considering required time for simulating each trial process solution. There are 6 objectives in this problem which the Pareto optimal for them is presented in two dimensional graphs to illustrate their trends versus each other. The total number of solutions in the Pareto set was 129 out of which 69 solutions represent the alternative of DME process without recovery unit.

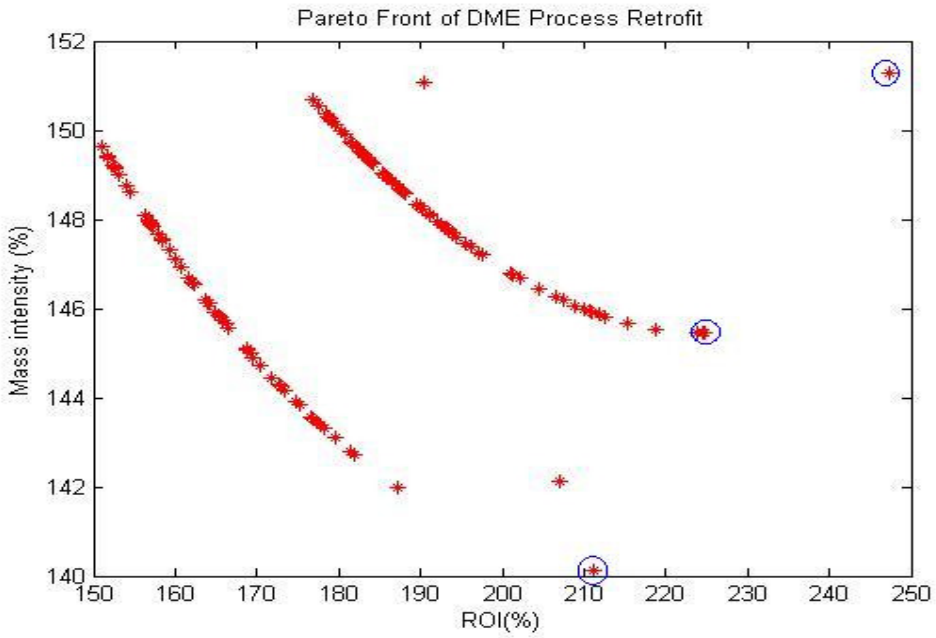


Figure 6.8- Approximate Pareto front of DME process retrofit, ROI vs. Mass intensity circled in blue

The results presented in the figure above demonstrate that there are at least three Pareto solutions circled in blue which none of them is inferior in terms of ROI and Mass intensity. It should be mentioned that ROI must be maximized while mass intensity is minimized. The solutions in the graph which seem to be dominated by the circled points are Pareto members which are presented as non-dominated solution because they are not inferior in terms of one objective other than ROI and Mass intensity. However, showing all the objectives is not possible.

Interestingly, Figure 6.8 presents that ROI as the indicator of economy improvement and mass intensity as the indicator of recovering material or resource usage are conflicting in some region. The conflicting region which generates the Pareto solutions in Figure 6.8 corresponds to different alternatives for DME process. Existence of the recovery unit decreases Mass intensity index while it increases ROI% of the process. The most economic desirable solution corresponds to the binary decision variables of y1= 0 and y_2 =1 which represents DME process without recovery unit. Thus, in terms of economy aspect adding the proposed recovery unit is not

recommended. However, most promising solution in terms of Mass intensity, one of resource usage indices, corresponds to a process with the recovery unit. The decision variables for the extreme solution points in terms of each objective are presented in Table 6.7. The best case optimized by Shadiya (2010) using Aspen plus based on only one objective, economy of process, is also presented in Table 6.7. The ROI% corresponding to the case reported by Shadiya (2010) is 198 which is inferior to many of Pareto solutions obtained in this work. For the presented results significan figures are kept to show the capability of the proposed algorithm in locating precise solutions with very small differences although significan figures should be ignored in engineering applications due to existing uncertatnies.

However, the overall trend in the solution objective area is not conflicting for majority of solutions. This means in a particular alternative of the process decrease in Mass intensity is the same direction with increase in ROI% due to increase in the desirable product.

Table 6.7- DV values for the extreme solutions in terms of ROI and Mass intensity

Decision variable	Max. ROI (%) DVs	Min. Mass intensity (%) DVs	Reported solution by Shadiya (2010)
y_1	0	1	1
r_1	0	1	1
y_2	1	0	0
r_2	1	0	0
Reactor Temperature (˚C)	195.4053	195 1 0	364
First column pressure (bar)	12.90	10.642	10.4
ROI %	247.2427	211.2448	198.011

The non-dominated solutions in terms of each pair of objective functions are shown in the subsequent graphs. The non-dominated points are circled in each graph.

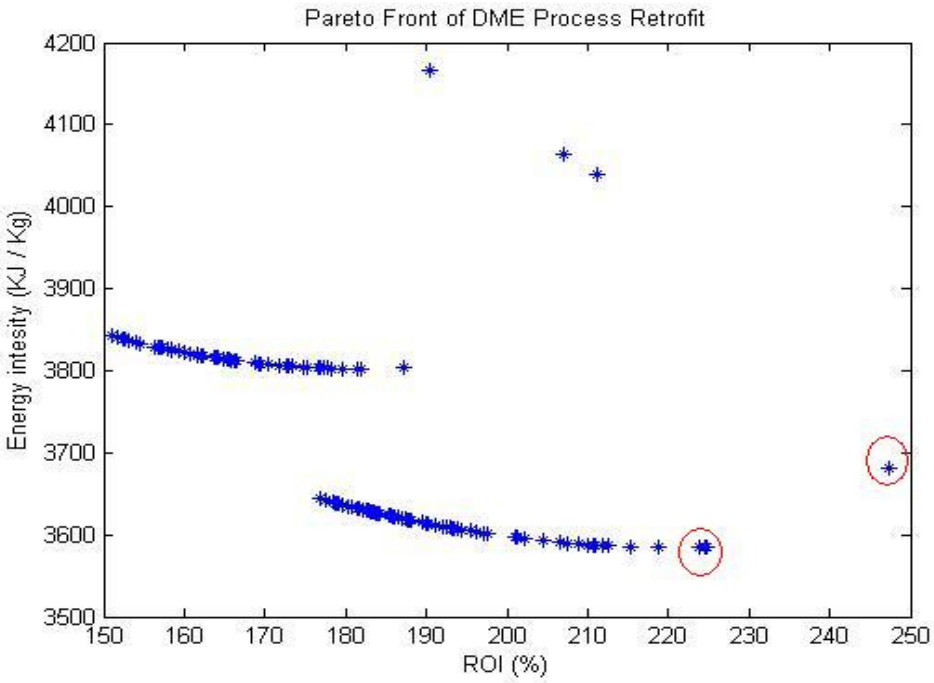


Figure 6.9- Approximate Pareto front of DME process retrofit, ROI and Energy intensity circled in red

The overall trend in the results shown in terms of ROI and Energy intensity, Figure 6.9, demonstrate that ROI increases when energy consumption in the unit is reduced. However, there is also conflict between Energy intensity index and ROI% in some regions. The reason for this is that although in the absence of the recovery unit the overall energy intensity may reduce for some operating conditions, the desired product amount reduces which results in larger energy intensity. Different duty of the reactor heating jacket and the heat exchangers after the reactor contributes to the observed trend decreasing energy intensity with increasing ROI% at some region.

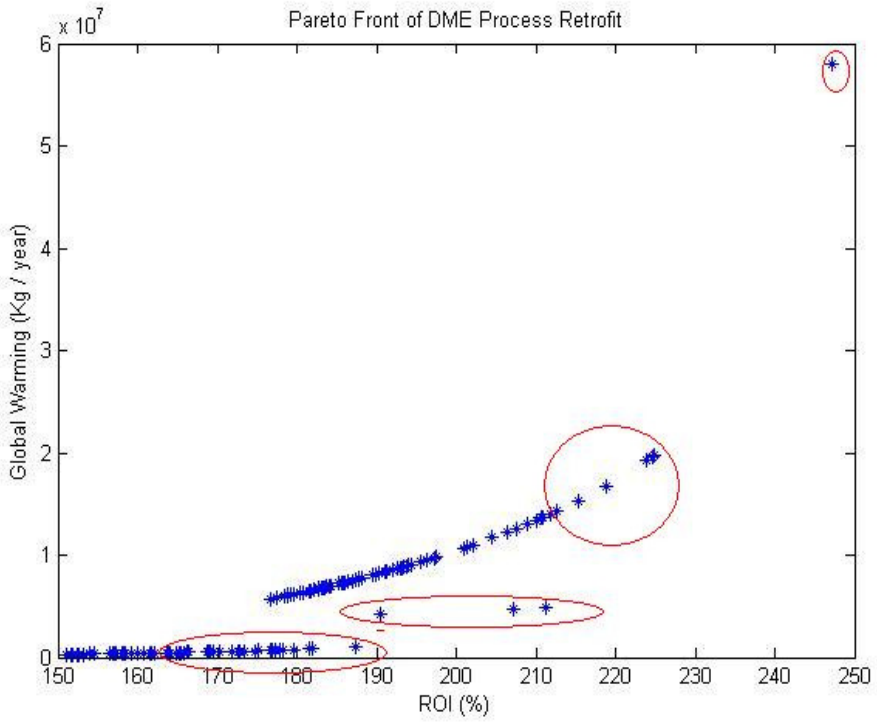


Figure 6.10- Approximate Pareto front of DME process retrofit, ROI% and Global warming circled in red

The results shown, Figure 6.10, in the graph above are in good agreement with the fact that adding a recovery unit reduces the emission of methanol to environment which causes global warming. The best solution in terms of ROI which corresponds to the process without recovery unit has the highest global warming index. Also again in this case variation of reactor temperature and the first distillation column in both process alternatives generates several Pareto solutions in between the extreme cases.

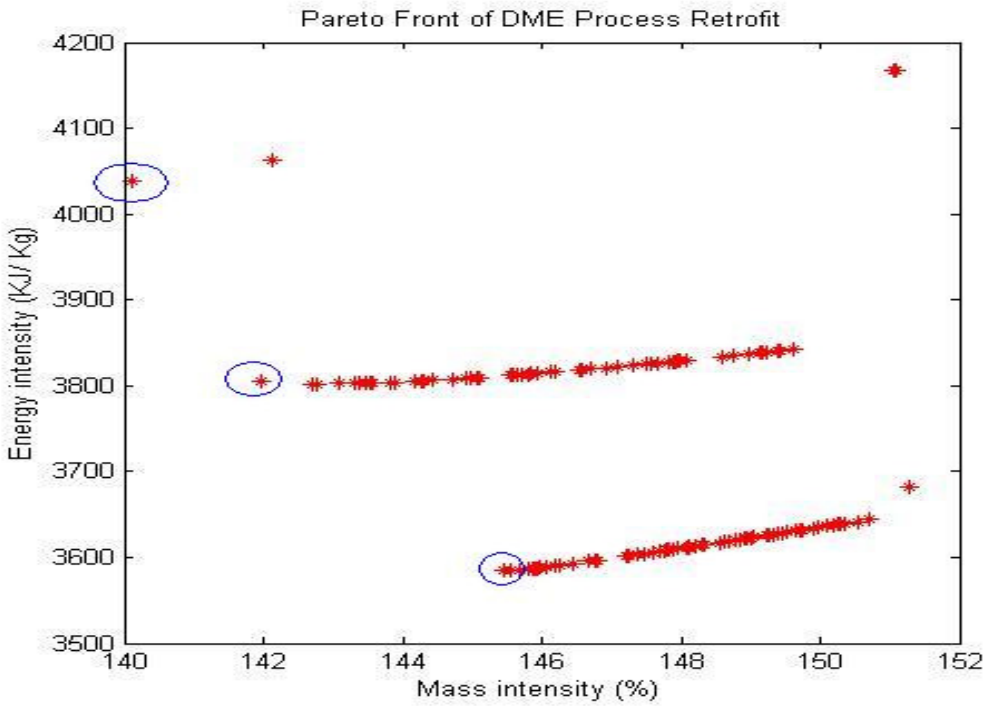


Figure 6.11- Approximate Pareto front of DME process retrofit, Mass intensity and Energy intensity circled in blue

Figure 6.11 shows the solutions in terms of Energy intensity and Mass intensity. The overall trends in most regions shows that decrease in the production of the desired product increases Energy intensity and also increases Mass intensity. However, there are some conflicting regions for these two indices which represent increase in energy consumption in the process due to a combination of decision variables for the operating condition for each alternative which does not increase the production of desired product with a same rate. Variation of the reactor temperature and the first column pressure changes the conversion and separation of methanol which is in conflict with increases the total energy consumed in the process. In addition to this existence of the recovery unit increases the energy consumption while decreases the mass intensity in the process.

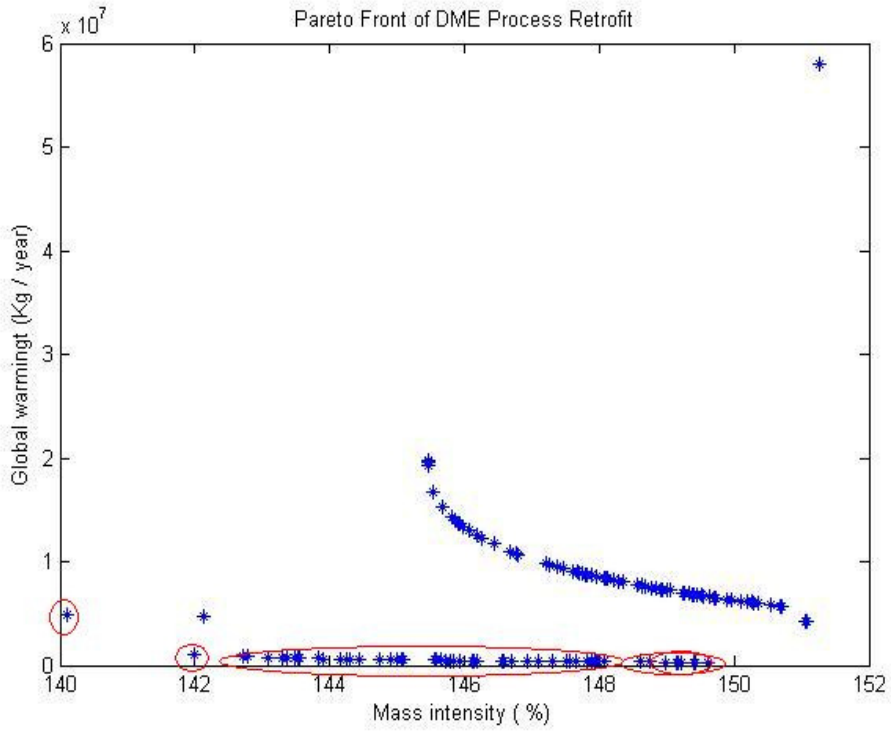


Figure 6.12- Approximate Pareto front of DME process retrofit, Mass intensity and Global warming circled in red

The Pareto optimal solutions in terms of Mass intensity and Global Warming indices are generated because of different contribution of Dimethyl ether and methanol in global warming. The Higher is the production of DME the less is Mass intensity index however, the higher is the production of DME the higher is Global warning index.

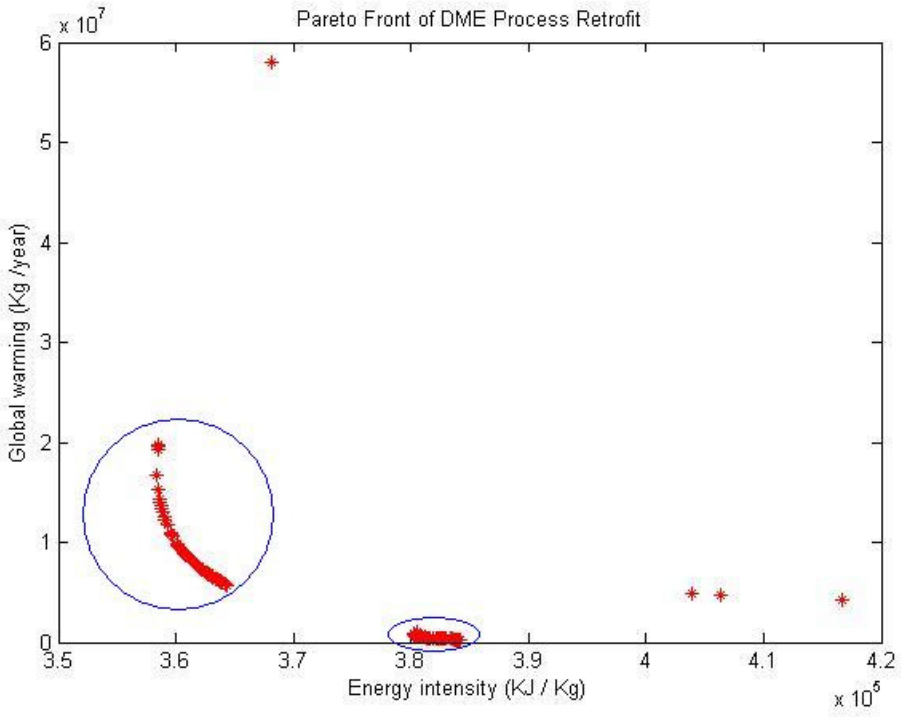


Figure 6.13- Appreximate Pareto front of DME process retrofit, Energy intensity and Global warming circled in blue

An obvious conflicting trend is observed in Figure 6.13 between Global warming and energy intensity. This is due to the fact that adding the recovery unit significantly decreases Global warming due to reduction in methanol emission. However, this increases the energy consumption in the process.

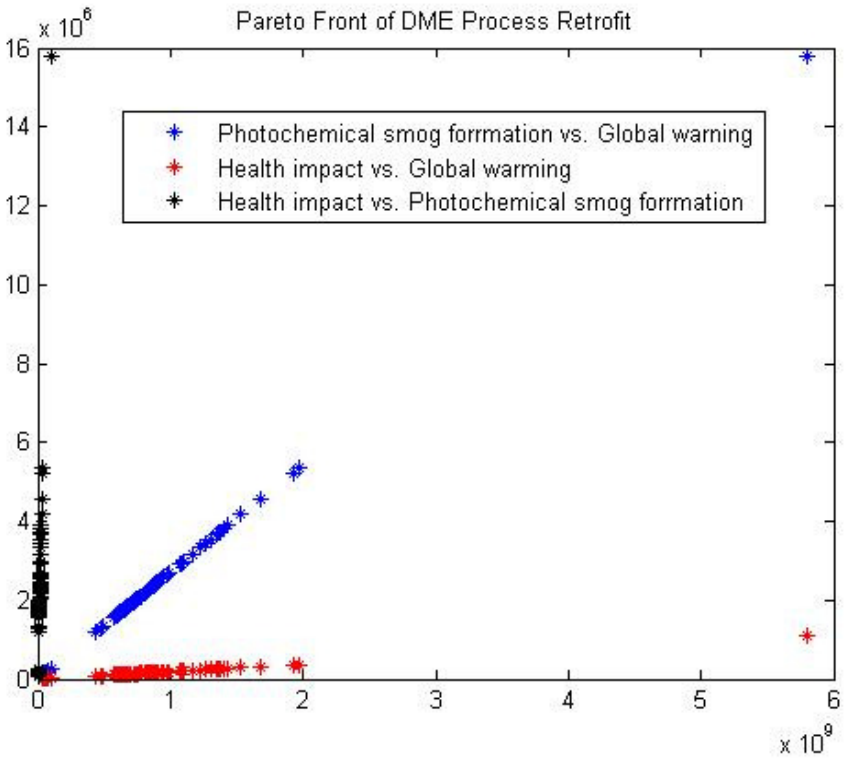


Figure 6.14- Approximate Pareto front of DME process retrofit, Energy intensity, Global warming and Health impact

Figure 6.14 shows three indices together, global warming, Photochemical smog formation and Health impact, which have a same trend and are not conflicting in any region. The difference in the slope is due to different potency factors for chemical contributing to each index. It *was* possible to minimize only one of these objectives to reduce the number of objectives of the problem if a prior knowledge about this fact was available. Sensitivity analysis on these objectives and decision variables would have helped to obtain this knowledge but it would take longer time and more effort than applying the framework without incorporating any type of priori knowledge on the objective function which might cause missing some conflicting region in some cases.

6.6.6.2 Results of DME Process Design

DME process retrofit Pareto optimal solutions obtained using the proposed framework were presented and discussed in previous section. However, as described in earlier chapters, a process has to be seen as an integrated unit for design synthesis and optimization. Therefore, in this case the size of reactor is also considered as a decision variable. This means that the base DME process has to be also changed if a different size is selected by the optimization framework. The other parameter that is considered here as a new decision variable in addition to decision variables for previous case is the fresh feed mass flow rate to the process. Two inequality constraints are added to the formulation of optimization problem (Equations 6.16 and 6.17) to enforce the demand of market. The decision variables for this case and their corresponding range are presented in Table 6.8.

Table 6.8-Rrange of decision variables

Decision Variable (DV)	Range
Reactor temperature (˚C)	195-400
Distillation tower pressure(bar)	10.50 to 14
Flow ratio of stream going to recovery unit	0.1 to 1
Flow ratio of stream going to waste treating facility (stream waste1)	0.2 to 1
Fresh feed mass flow (Kg/hr)	7000 to 15000
Reactor volume (m^3)	0.05 to 8

The ultimate goal of implementing the proposed algorithm is to select the most desirable operating conditions and configuration of DME process presented in the superstructure. As

described the superstructure includes two alternatives for this process: with methanol recovery unit and without recovery unit.

The framework was applied on this problem and found 118 Pareto optimal solutions within 30 minutes using 50 particles in the swarm. The algorithm stopped at iteration number 47. Out of 118 solutions in the Pareto solution set 58 solutions correspond to the cases which do not have the methanol recovery unit.

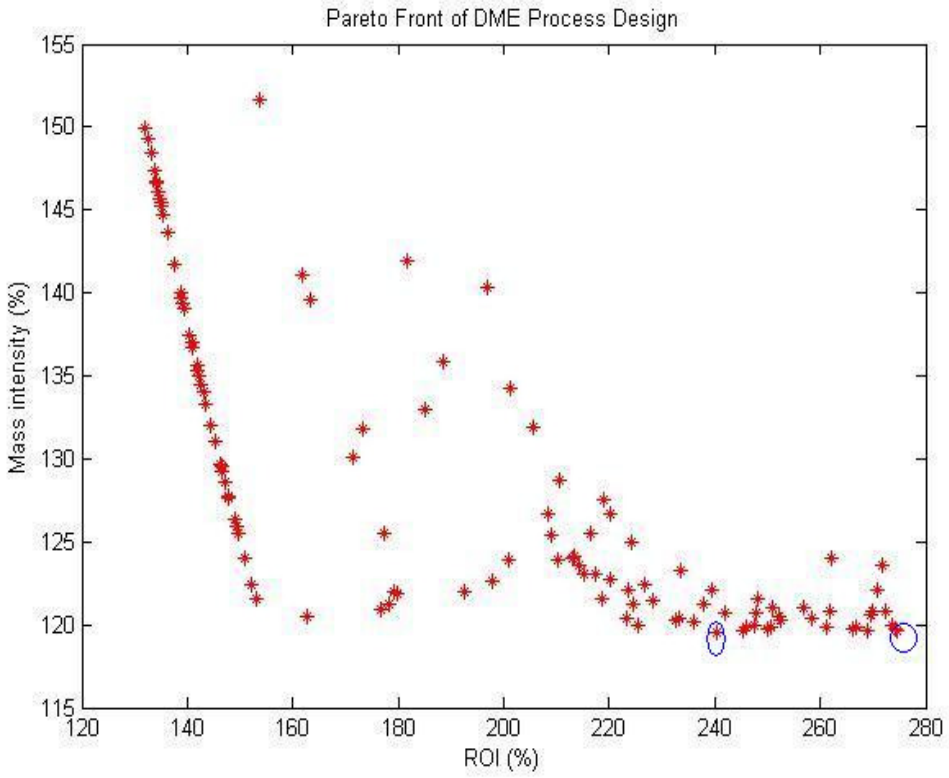


Figure 6.15- Approximate Pareto front of DME process design, ROI vs. Mass intensity circled in blue

Result in Figure 6.15 for Mass intensity and ROI % of the process hardly show few Pareto solutions. This is because of very small difference between Mass intensity indexes of Pareto solutions. In this case both Pareto solutions correspond to the alternative without methanol recovery unit. In both cases which are shown by blue circles in Figure 6.15 mass intensity are almost the same with a slight difference however due to different temperature and feed mass flow

rate the ROI% differs noticeably between two cases. The decision variables for these cases are presented in Table 6.9. Unlike the retrofit case, in the design case of DME since the least Mass intensity index correspond to the case which there is no recovery unit in the process. This is happens because in this case the reactor size is varied thus it can be designed large enough to decrease unconverted methanol so that there is no need for recovery unit. On the other hand best ROI % happens when the reactor is smaller. That reduces the heat duties and reactor cost and can be compensated by a slight increase in the fresh feed.

Table 6.9- Decision variables value corresponding to the extreme solutions in terms of ROI and Mass intensity

Decision variable	Min. Mass intensity (%) DVs	Max. ROI (%) DVs
y_1	0	0
r_1	0	0
y_2	1	1
r_2	1	1
Reactor Temperature (˚C)	244.9154	254.1630
First column pressure (bar)	11.472	10.516
Reactor volume (m^3)	1.8523	0.2000
Fresh feed mass flow (Kg/hr)	1.0003e+004	1.0815e+004
ROI %	240.2973	274.5669
Mass intensity (%)	1.1953	1.1968

The non-dominated solutions in terms of each pair of objective functions are shown in the subsequent graphs. The non-dominated points are circled in each graph.

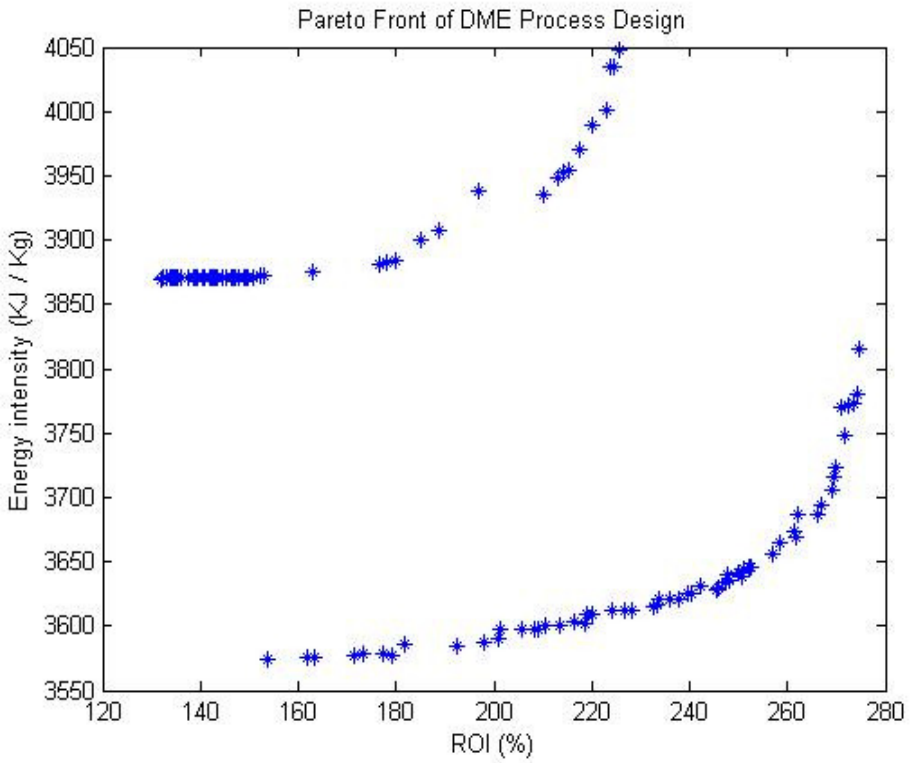


Figure 6.16- Approximate Pareto front of DME process retrofit, ROI% and Energy intensity shown in the lower curve

There lower curve shown in Figure 6.16 presents the Pareto front in terms of ROI% and Energy intensity indices. There is an expected trend of increase in ROI with decrease in energy consumption of the process. The least energy consumption happens in absence of the methanol recovery unit with the smaller amount of fresh feed entering to the process which reduces the heat duty of heat exchanger and reactor. Larger reactors also help to keep the production of DME high which reduces Energy intensity even more. As fresh feed and temperature of reactor increase the production of DME increases which in turn increases the ROI of process while the ratio of energy consumption over desired product becomes larger. The extreme solutions in terms of energy intensity and ROI% are presented in Table 6.10.

Table 6.10- Decision variables value corresponding to the extreme solutions in terms of ROI and Energy intensity

Decision variable	Min. Energy intensity (%) DVs	Max. ROI (%) DVs
y_1	0	0
r_1	0	0
y_2	1	1
r_2	1	1
Reactor Temperature (˚C)	195	254.1630
First column pressure (bar)	12.90 0	10.516
Reactor volume (m^3)	8	0.2000
Fresh feed mass flow (Kg/hr)	7.7095e+003	1.0815e+004
ROI %	153.6696	274.5669
Energy intensity (KJ /Kg)	3.5748e+003	3.8157e+003

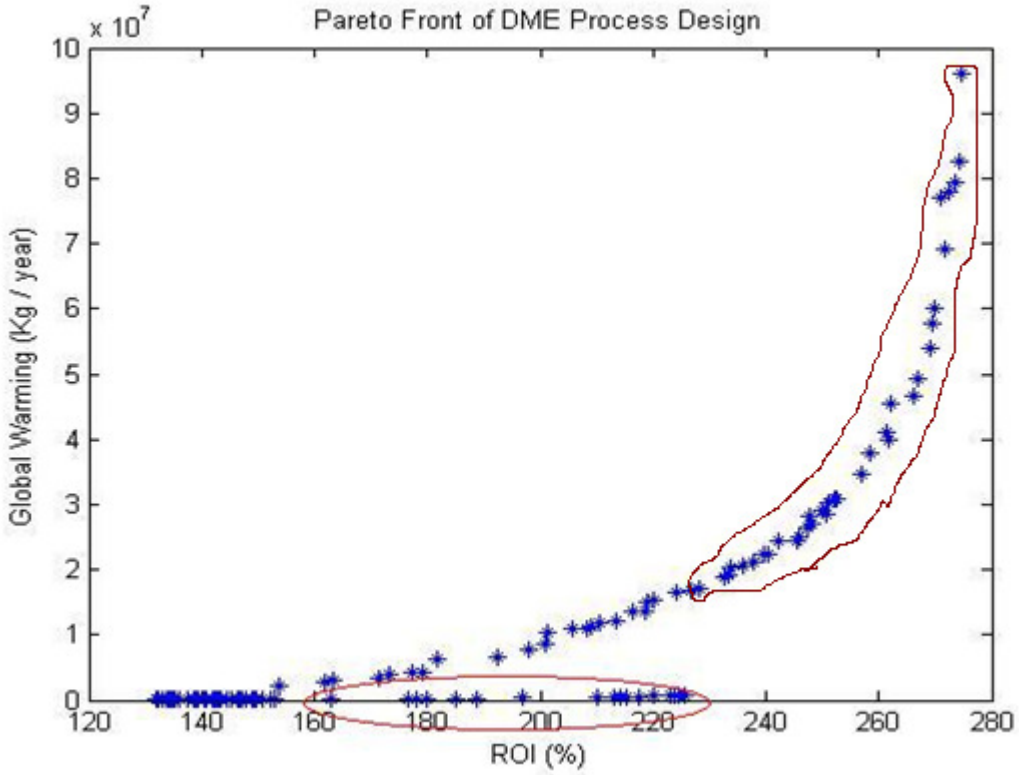


Figure 6.17- Approximate Pareto front of DME process design, ROI and Global warming circled in red

The Pareto solution set in terms of ROI% and Global warming index is shown in Figure 6.17. The solution in the red circle correspond to the alternative which includes the recovery unit for methanol and also as it can be seen ROI% for these point are relatively low due to smaller amount of fresh feed. In addition, the reactor size for these solutions are relatively large so that the conversion of methanol is kept high which in turn reduces the emissions to the environment. However, this would cause much more cost and energy consumption in the process.

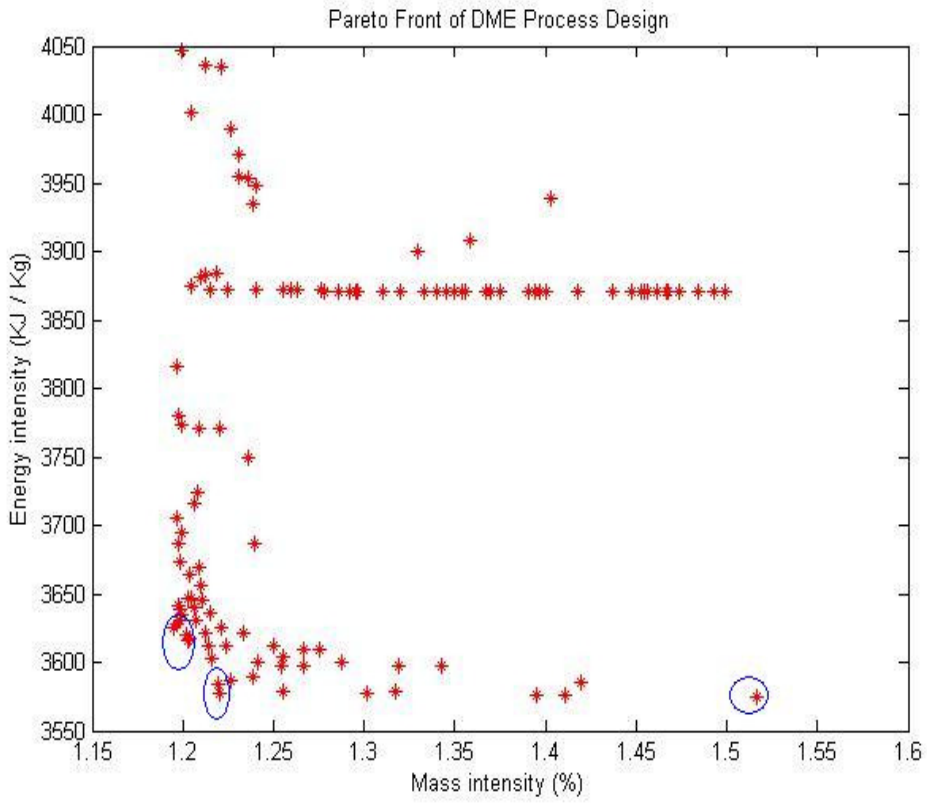


Figure 6.18- Approximate Pareto front of DME process design, Mass intensity and Energy intensity circled in blue

Figure 6.18 shows the solutions in terms of Energy intensity and Mass intensity. The Pareto solutions in this figure correspond to the alternatives without methanol recovery unit. However, solutions which have better mass intensity also have higher rate of fresh feed and higher temperature in the reactor which results in more production of DME and thus reducing Mass

intensity index. On the other hand solutions which are better in terms of Energy intensity handle less fresh feed in the process to keep the energy consumption low and also they require larger reactors with lower operating temperature.

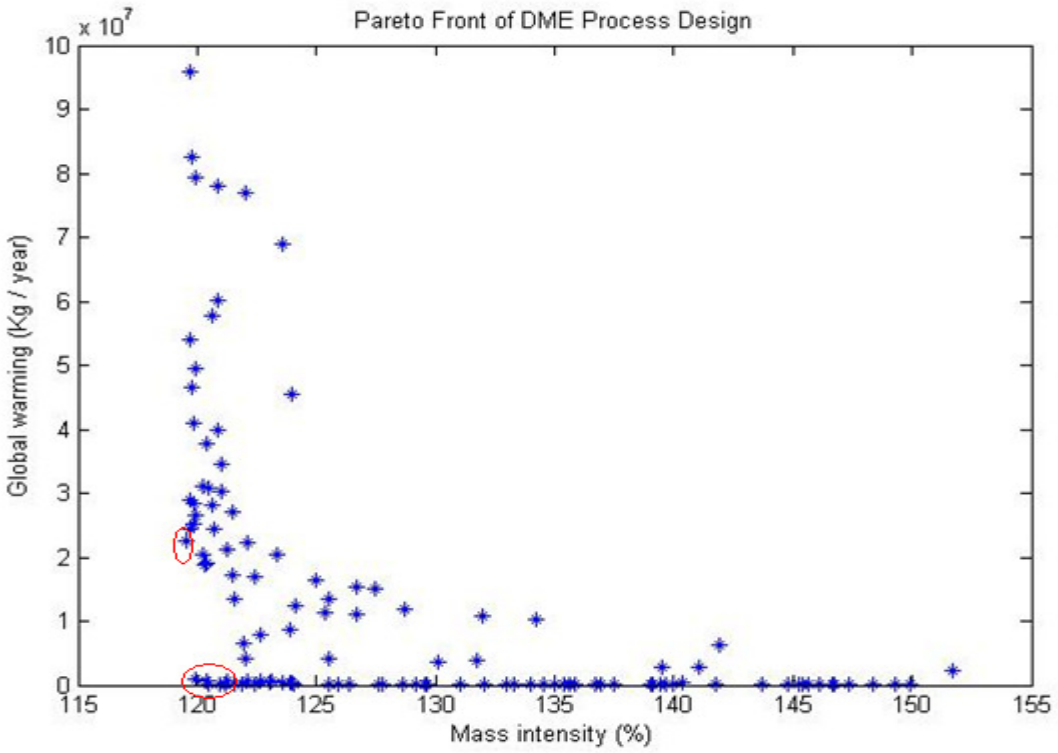


Figure 6.19- Approximate Pareto front of DME process retrofit, Mass intensity and Global warming circled in red

The Pareto optimal solutions in terms of Mass intensity and Global Warming indices are presented in Figure 6.19. Although, they are not clearly distinct in the plot, the Pareto optimal solutions in terms of these two objectives correspond to different process alternative. DME process with the methanol recovery unit reduces emissions to the environment thus it is more desirable in terms of Global warming index. The extreme solutions in terms of these two indices are presented in Table 6.11.

Table 6.11- Decision variables value corresponding to the extreme solutions in terms of Global warming and Mass intensity

Decision variable	Min. Mass intensity (%) DVs	Max. ROI (%) DVs
y_1	0	1
r_1	0	1
y_2	1	0
r_2	1	0
Reactor Temperature (˚C)	244.9154	195
First column pressure (bar)	11.472	1290
Reactor volume (m^3)	1.8523	8
Fresh feed mass flow (Kg/hr)	1.0003e+004	7.7755e+003
Global warming (Kg / year)	2.2491e+007	2.3174e+004
Mass intensity (%)	1.1953	1.4989

Due to very small difference between Pareto solutions in terms of Global warming and Energy intensity plotting them in a two dimensional graph would not help to further understand the trend. However, Figure 6.21 depicts two extreme solutions in terms of Global warming and Energy intensity.

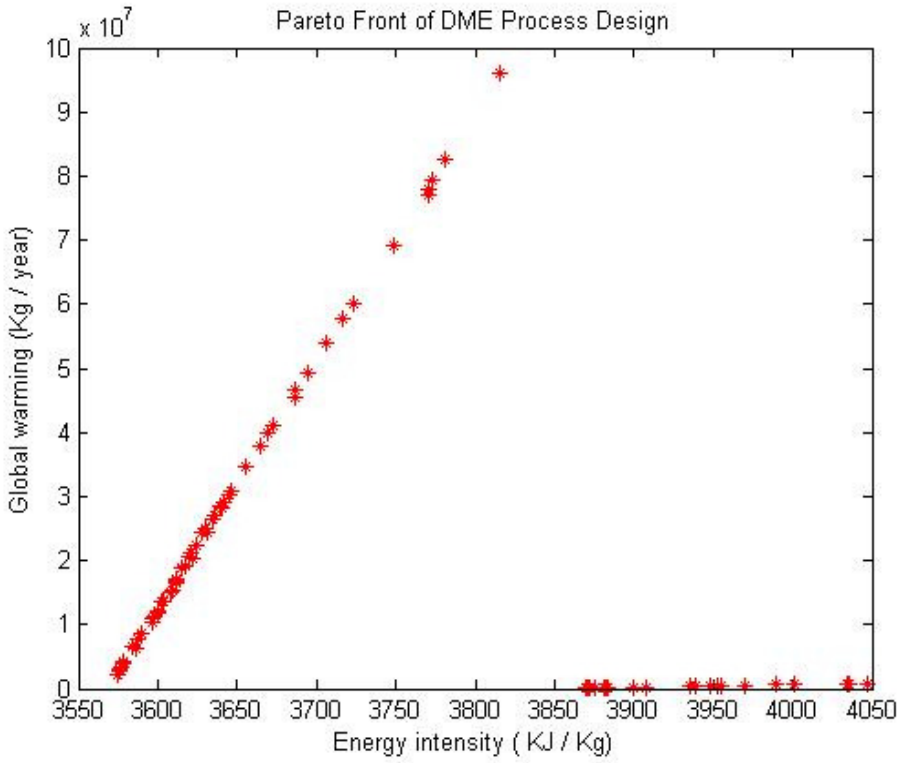


Figure 6.20- Approximate Pareto front of DME process design, Energy intensity and Global warming

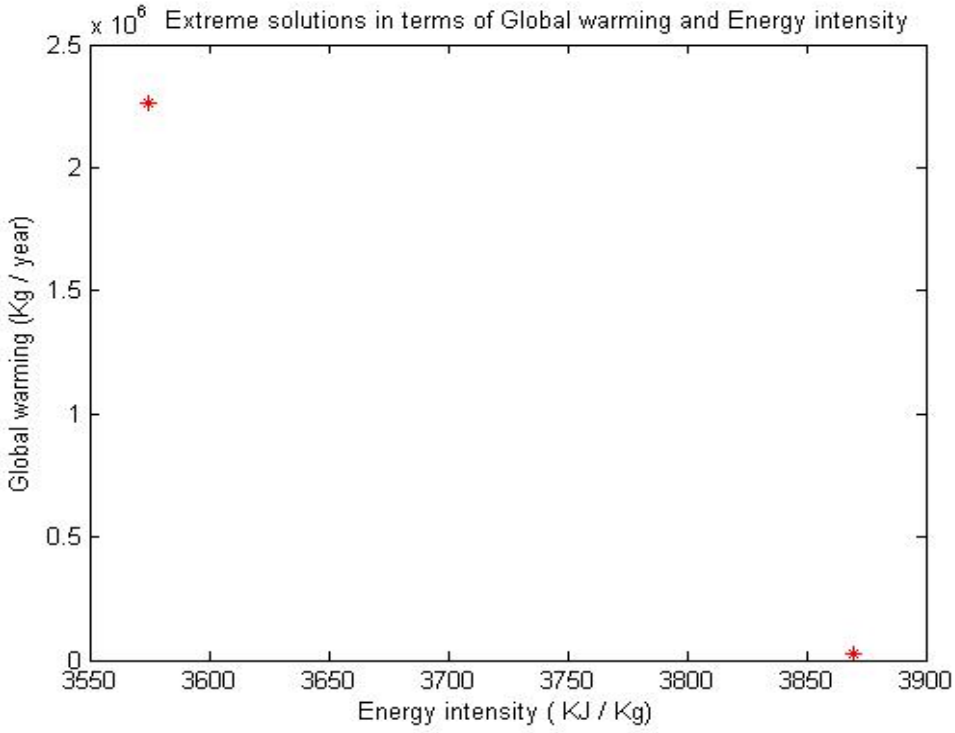


Figure 6.21- Extreme Pareto solutions in terms of Energy intensity and Global warming

Figure 6.22 shows three indices together, global warming, Photochemical smog formation and Health impact, which have a same trend and are not conflicting in any region. The difference in the slope is due to different potency factors for chemical contributing to each index. It was possible to minimize only one of these objectives to reduce the number of objectives of the problem if a prior knowledge about this fact was available. Sensitivity analysis on these objectives and decision variables would have helped to obtain this knowledge but it would take longer time and more effort than applying the framework without incorporating any type of priori knowledge on the objective function which might cause missing some conflicting region in some cases.

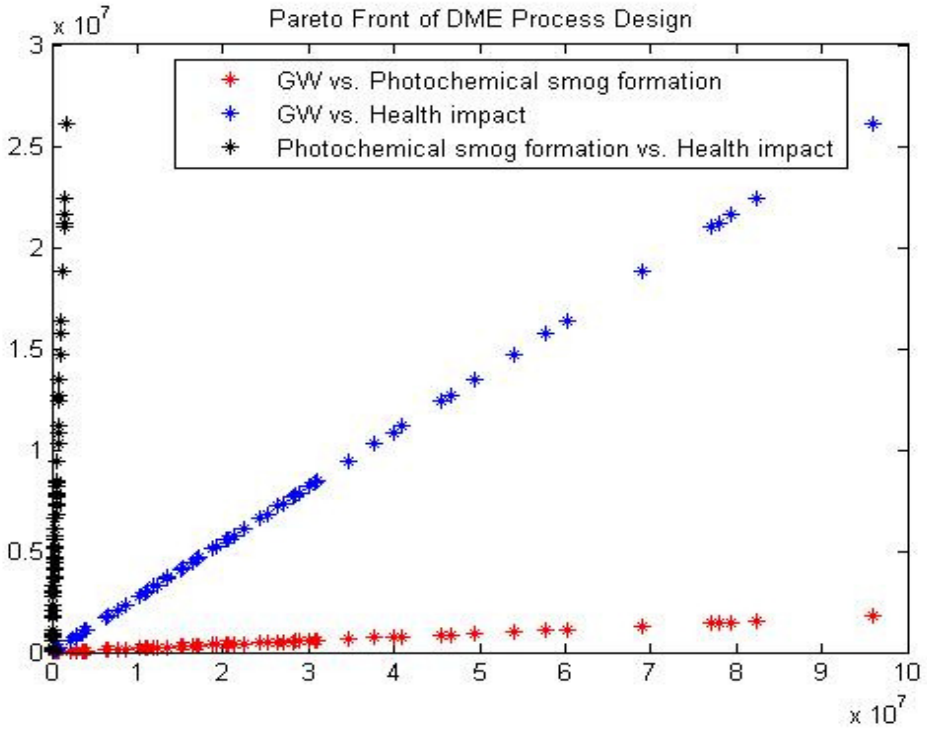


Figure 6.22- Approximate Pareto front of DME process design, Energy intensity, Global warming and Health impact

6.6.7 Selecting a Final Solution using Decision Making Module

The Pareto optimal solution set for both DME process optimization cases have been discussed in previous sections. However, due to existence of many solutions in the final solution sets and also high dimensionality of objective space it is hard and almost impossible to select a single desirable solution without high cognition load. Therefore, the fuzzy ranking method explained in earlier sections of this chapters is usedto extract decision makers' preferences based on the available solution in the Pareto sets and select a final solution desired by decision makers.

6.6.7.1 Final Solution of DME Design Case

The maximum and minimum value of each objective function in the Pareto solution set obtained for DME design case, Table 6.12, was presented to decision makers. The compromised range for each objective selected by decision makers is presented in table 6.13. The preferred ranges presented in Table 6.13 are examples to show the applicability of the methodologies and may not have any specific justification behind them.

Table 6.12-Maximum and minimum value of objectives in the Pareto set obtained for DME design case

Objective function	Max Value	Min Value
ROI%	274.5669	132.1108
Mass intensity (%)	1.5167	1.1953
Energy intensity (KJ / Kg)	4.0471e+003	3.5748e+003
Global warming (Kg/year)	9.5907e+007	2.3174e+004
Photochemical smog formation (Kg / year)	1.7883e+006	432.2975
Health impact	2.6108e+007	6.2983e+003

Table 6.13- Compromised range inserted by decision makers for objectives of DME design

Objective function	Desired Max	Compromised Min
- ROI%	-190	- 230
Mass intensity (%)	1.3	1.45
Energy intensity (KJ / Kg)	3,600	3,800
Global warming (Kg/year)	80,000	500,000
Photochemical smog formation (Kg / year)	1,000	50,000
Health impact	10,000	300,000

Application of the decision making module on the Parto solution of DME design case resulted in the final solution presented in Table 6.14. In order to strictly select a solution according to the compromised range presented in Table 6.13, Product fuzzy method was selected for calculating the aggregated rank of each solution. Based on the presented range no solution was found to be strictly according to the decision makers' preferences. Therefore, the upper limit of one or more objective functions must be increased.

However, applying the Convex fuzzy method which is for selecting the closest solution to the decision makers' preferences resulted in the selection of the 52^{th} solution in the Pareto set whose objectives and decision variable values are presented in Table 6.14 and 6.15 respectively. It should be mentioned that the weighting factors in Equation 6.7 for calculation of aggregated rank using convex fuzzy method were all set to 1. Thus, there was no bias in assigning the fuzzy membership for any objective function.

Table 6.14-Objective values for the final solution (52th solution in the Pareto set)

Objective function	Value
ROI%	179.8688
Mass intensity (%)	1.2188
Energy intensity (KJ / Kg)	3.8841e+003
Global warming (Kg/year)	1.2350e+005
Photochemical smog formation (Kg / year)	2.3024e+003
Health impact	1.9307e+004
Temperature index	4

Table 6.15-Decision variables of the final solution (52th solution in the Pareto set)

Decision variable	Min. Mass intensity (%) DVs
y_1	1
r_1	1
y_2	0
r_2	0
Reactor Temperature (˚C)	195
First column pressure (bar)	12.90
Reactor volume (m^3)	3.0059
Fresh feed mass flow (Kg/hr)	9.5639e+003

All the sustainability indices are presented in following figure and tables separately for the final selected solution (solution 52) for DME design case.

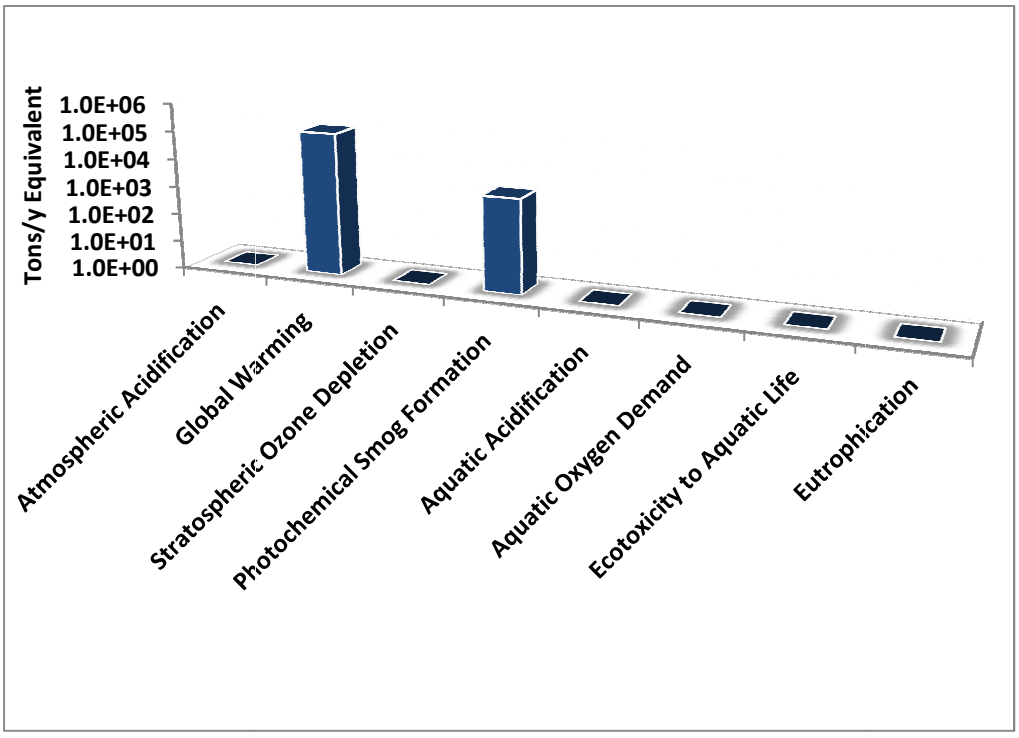


Figure 6.23-Environmental impact for final selected DME design solution

Table 6.16-Resource usage indices for final DME design solution

OUTPUTS for Resource Usage Evaluation		
Effective Mass Yield	82%	
E-Factor	0.4	Kg/Kg
Atom Economy	144%	
Mass Intensity	1.24	Kg/Kg
Mass Productivity	82%	
Reaction Mass Efficiency	82%	
Material Intensity	0.22	Kg/Kg
Energy Intensity/ Fossil Fuel Usage	1.6665	KW/Kg
Water Intensity	0.0	Kg/Kg

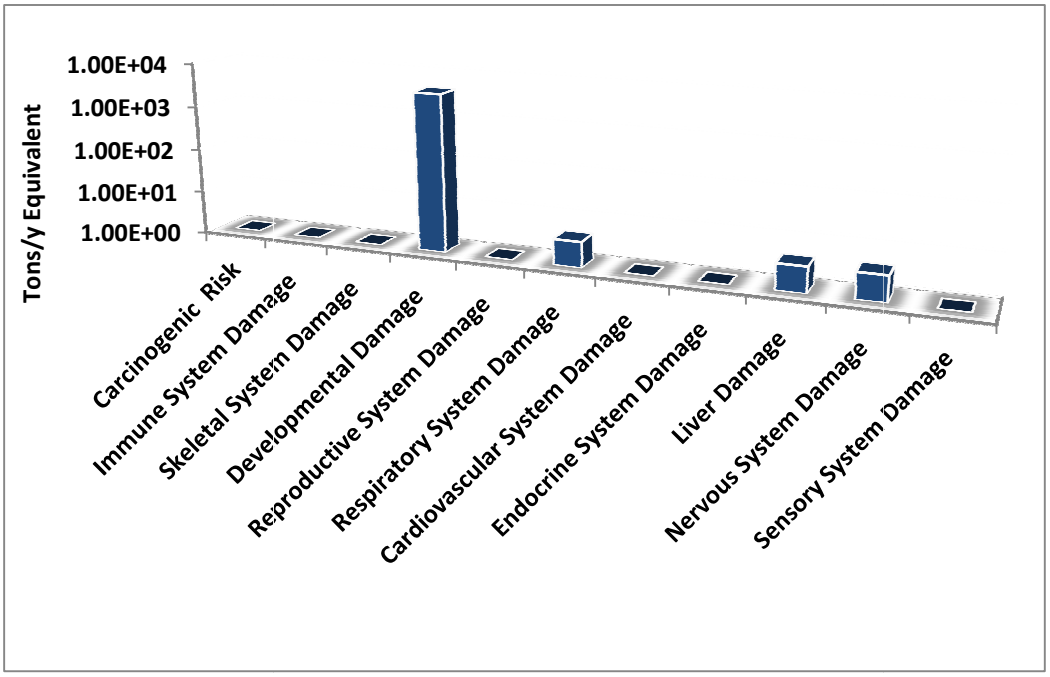


Figure 6.24-Health indices for final selected DME design solution

Table 6.17-Safety indices for final DME design solution

OUTPUTS for Process Safety Evaluation		
	Results	Maximum Value
Heat of main reaction index	0	0%
Heat of side reaction index	0	0%
Flammability index	8	100%
Explosiveness index	4	50%
Toxic Exposure Index	12	40%
Corrosiveness index	4	100%
Temperature index	4	50%
Pressure index	2	25%
Equipment safety index	4	50%
Safety Level of Process Structure index	2	20%
Total Inherent Safety index	40	

6.6.72 Final Solution for DME Retrofit Case

Likewise the DME design case both fuzzy methods were applied for selection a final solution among the Pareto solutions obtained for DME retrofit case study. The range of each objective function value and compromised range inserted by decision makers are presented respectively in Tables 6.18 and 6.19.

Table 6.18-Mmaximum and minimum value of objectives in the Pareto set obtained for DME retrofit case

Objective function	Max Value	Min Value
ROI%	247.2427	151.0984
Mass intensity (%)	1.5127	1.4012
Energy intensity (KJ / Kg)	4.1671e+003	3.5842e+003
Global warming (Kg/year)	5.7984e+007	2.9034e+005
Photochemical smog formation (Kg / year)	1.0814e+006	5.4130e+003
Health impact	1.5775e+007	7.9080e+004

Table 6.18-Ccompromised range inserted by decision makers for objectives of DME design

Objective function	Desired Max Value	Compromised Min Value
- ROI%	-180	- 220
Mass intensity (%)	1.45	1.5
Energy intensity (KJ / Kg)	3,750	4000
Global warming (Kg/year)	450000	2000000
Photochemical smog formation (Kg / year)	9000	500000
Health impact	120000	4000000

The objective functions values and decision variable values of the final solution obtained by applying the fuzzy product method is presented respectively in Tables 6.20 and 6.21.

Table 6.20-Objective values for the final solution (103th solution in the Pareto set)

Objective function	Value
ROI%	187.3152
Mass intensity (%)	1.4198
Energy intensity (KJ / Kg)	6.6692e+003
Global warming (Kg/year)	9.7899e+005
Photochemical smog formation (Kg / year)	1.8247e+004
Health impact	2.6686e+005
Temperature index	4

Table 6.21-Decision variables of the final solution (103th solution in the Pareto set)

Decision variable	Min. Mass intensity (%) DVs
y_1	1
r_1	1
y_2	0
r_2	0
Reactor Temperature (˚C)	196.5108
First column pressure (bar)	11.211

The result obtained by the fuzzy product method for selecting the final solution is strictly according to the desirable range inserted by decision makers. It should be noted that the compromised ranges in this example were wider to show the application of product fuzzy method as a strict method of selection. In addition the convex fuzzy method was applied to explore the differences between these two methods. The same compromised ranges were used for this method also and all the weighting factors considered one to avoid any bias toward specific objective functions. Interestingly as it was expected, both methods found exactly the same solution. In the cases that all objectives of at least one Pareto solutions fall in the compromised ranges, both methods result in a same solution. In addition, all the sustainability indices are presented in the following tables and figures separately for the final DME retrofit solution.

Table 6.22-Resource usage indices for final DME retrofit solution

OUTPUTS for Resource Usage Evaluation		
Effective Mass Yield	70%	
E-Factor	0.1	Kg/Kg
Atom Economy	144%	
Mass Intensity	1.42	Kg/Kg
Mass Productivity	70%	
Reaction Mass Efficiency	70%	
Material Intensity	0.4	Kg/Kg
Energy Intensity/ Fossil Fuel Usage	1.8525	KW/Kg
Water Intensity	0.0	Kg/Kg

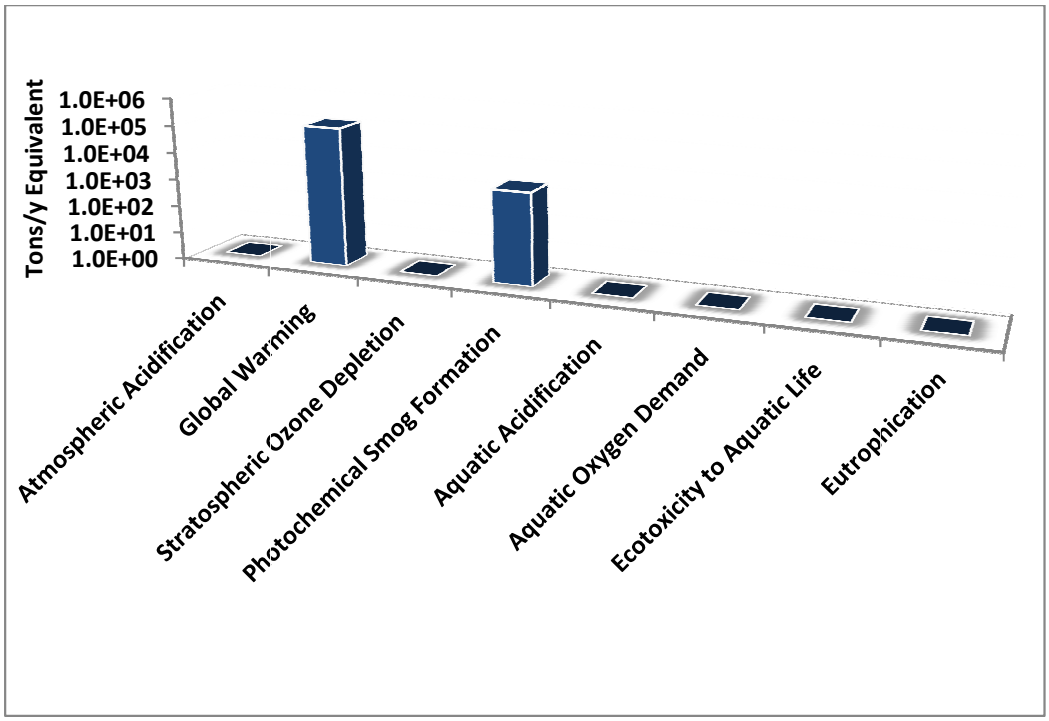


Figure 6.25-Environmental indices for final DME retrofit solution

Table 6.23-Safety indices for final DME retrofit solution

OUTPUTS for Process Safety Evaluation		
	Results	Maximum Value
Heat of main reaction index	0	0%
Heat of side reaction index	0	0%
Flammability index	8	100%
Explosiveness index	4	50%
Toxic Exposure Index	12	40%
Corrosiveness index	4	100%
Temperature index	4	50%
Pressure index	2	25%
Equipment safety index	4	50%
Safety Level of Process Structure index	2	20%
Total Inherent Safety index	40	

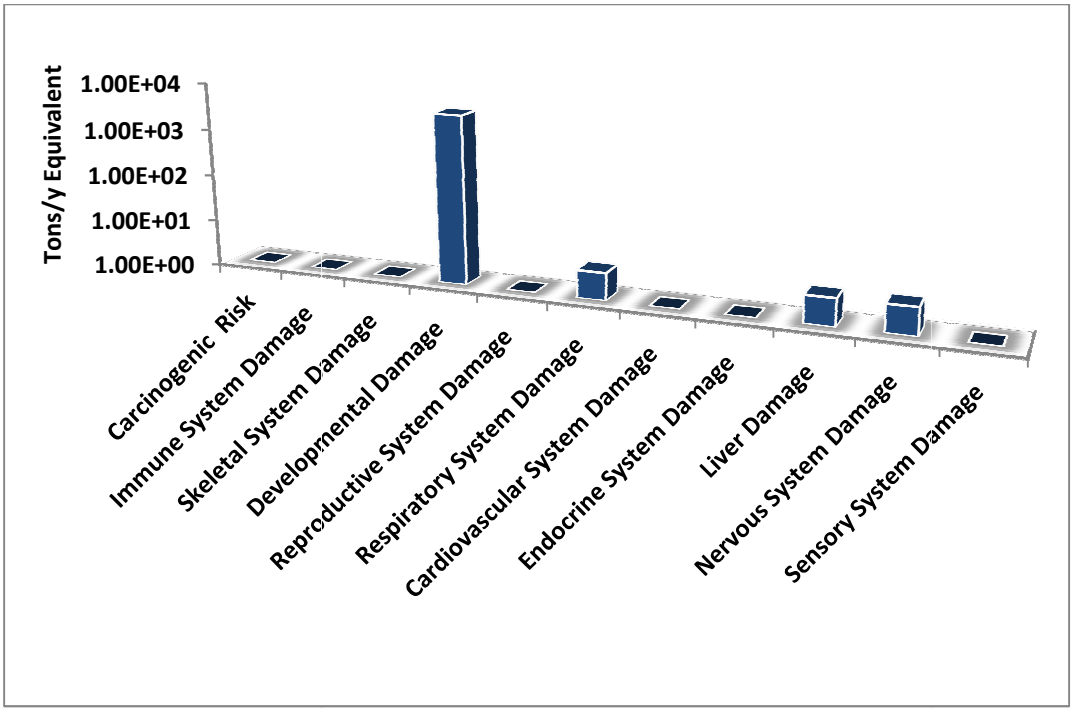


Figure 6.26-Health indices for final DME retrofit solution

CHAPTER 7

CONCLUSION AND FUTURE WORK

7.1 Research Contributions

With increasing negative impacts of industrial activity on environment and society, traditional criteria for design which only addresses economic aspects of processes are no longer appropriate. The sustainability of the processing industry has gained attention globally. There is increasing pressure to become more environmentally friendly and socially acceptable. How do companies consider new green technology and determine how to incorporate this new technology into new or existing processes? How do companies choose from among several potentially sustainable alternatives?

During past decades many researchers have contributed to incorporate sustainability into process design and answer the above questions. However, several challenges still exist in front of engineers in chemical industries for integrating the sustainability concerns into design. Mainly, these challenges arise due to the existence of many metrics of sustainability which neither can be compromised over each other nor can be scalarized and combined into a single criterion. In addition, due to the complexities in chemical process design which adds up to formulation of sustainability concerns for design there is no comprehensive tool available to enable engineers to reliably design for sustainability.

Accordingly, incorporation of sustainability in real industrial designs needs methodologies and frameworks which have several features:

- The most important feature of a framework is the reliability of it in terms of design. Industrial sectors will not apply a tool or method which is based on surrogate or simplified methods.
- The framework has to enable incorporation of several criteria without imposing a large effort on process engineers. Very complex methodologies which require a lot of mathematical formulation and simplification of models are not convenient for regular process engineers.

In this research, an optimization based methodology was developed in the form of a computational framework to consider sustainability criteria in process design. This methodology addresses the above problems with special intention to chemical process design. Through this methodology, sustainability is mathematically formulated into a set of objective functions, each of which is constructed in a rigorous scientific manner. A well-defined methodology that includes process optimization under social, economic and environmental constraints was developed. This methodology was built upon robust and modified multi-objective algorithms that handle selected sustainability criteria and constraints. This project developed a rigorous evolutionary based framework in which the optimum goal is to develop a multi-objective optimization and decision making methodology to handle multiple criteria such as sustainability metrics by exploring all feasible process alternatives using the chemical process simulator, Aspen Hysys, and the comprehensive metrics for social, environmental and economic aspects of sustainability.

Therefore, a systematic framework for optimal process design considering sustainability constraints shown in Figure 7.1 was developed. In this framework, the MLMOPSO optimization algorithm interacts with a sequential modular process simulator to find the optimum configuration and operational conditions of a given process. At each step the optimizer communicates decision

variables (DVs) with a process simulator (Aspen Hysys here) which simulates the process based on new DVs. The results of the simulator which are very realistic are used to evaluate the objective functions and are passed to the optimizer again. In order to design for sustainability the evaluation of objective functions is performed a sustainability evaluator module which is programmed in MATLAB. The results of using our methodology on that process were compared with a base case. The criterion for success was that our methodology shown significant sustainability improvement compared to the base case and the proposed framework is suitable to any process.

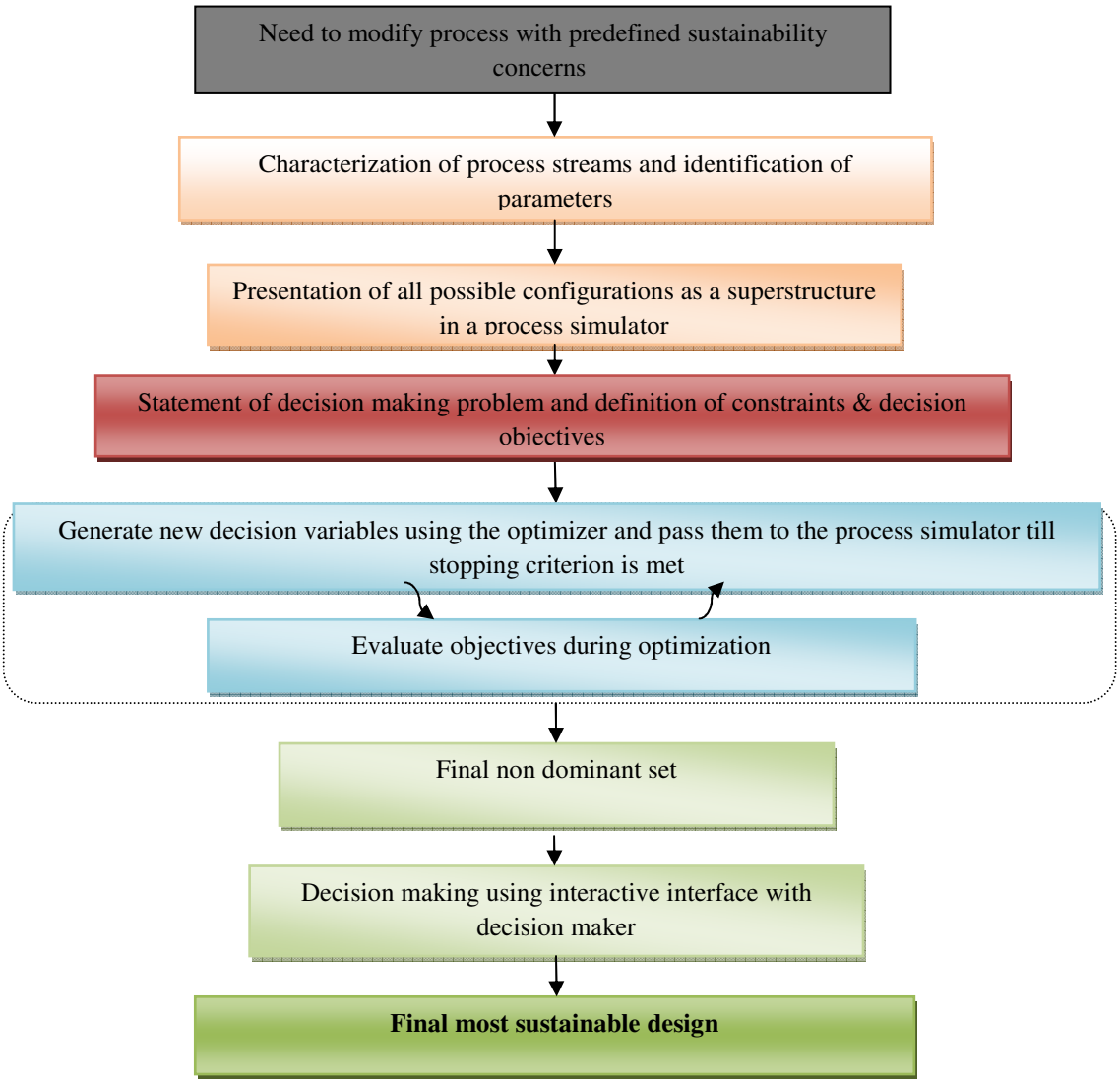


Figure 7.1, Flow diagram for overall methodology

This research contributes to sustainability development in process design. This project addressed a major challenge in process design as engineers will now be able to design for multiple criteria, not restricted but specifically sustainability concerns. This work developed approaches for the optimization of chemical processing technologies with sustainability considerations. The contributions of this research are:

- Development of a modified Particle Swarm algorithm to handle a multitude of objectives in mixed integer decision making spaces.
- Formulation of an algorithm that incorporates multi-objective optimization using a non-dominated methodology to enable process design with several objectives especially to satisfy social, economic and environmental objectives.
- Development of a computer aided tool to assess sustainability metrics in tandem with a process simulator and a robust multi-objective PSO algorithm to synthesize and explore all the feasible process alternatives with respect to economic, environmental and social impacts.
- A decision making strategy was proposed for selecting the most sustainable process from multiple optimum alternatives that will remain sustainable in the future.
- Evaluation of performance of the methodology by testing it on an industrial process case study.

7.2 Implication of Results and Broader Impacts

In order to incorporate all sustainability metrics in complex process design problems a novel stochastic optimization algorithm called Multi-Leader Multi-Objective Particle Swarm Optimization (MLMOPSO) was developed and introduced in this work. This algorithm is a robust optimization algorithm for handling a multitude of optimization objectives in mixed integer decision making spaces. Also strategies for altering the parameters associated with

MLMOPAO algorithm were introduced which are novel from the aspect of being independent from the maximum number of iterations or any other pre-specified parameter which requires pre knowledge about the functionality of the problem in hand. In addition, a new combined strategy was introduced and incorporated for handling optimization constraints in nonconvex optimization problems which consist of two phase. Switching between constraint handling phases and termination of the algorithm were automated to decrease interference of the decision maker in the optimization procedure. It should be mentioned that the proposed algorithm and its parameter adaption strategies is quiet novel as they are able to handle both multi-objectives and mixed integer problems while the original PSO and its modified versions are not.

The implementation of the proposed self-adaptive algorithm on well-known benchmark optimization problems proved the capability of the algorithm in handling various types of optimization problems including single and multi-objective problems with highly non-convex and mixed integer search spaces. The proposed algorithm converged to the solution precisely with very reasonable computational effort which enables application of MLMOPSO algorithm for industrial scale problems. The introduced stochastic optimization algorithm is capable of handling various types of problems whereas, none of the existing optimization algorithms has been reported to be well suited for handling a combination of complexities such as mixed integer nonconvex search area with multiple objectives in more reasonable computational expense.

The results presented in this work imply that the proposed self-adaptive algorithm is applicable to a vast area of multi-criteria decision making problems dealing very complicated objectives and constraints and is not restricted to the tested benchmark functions. This broadens the application of this algorithm to engineering problems which almost always result in very complicated multi-criteria decision making problems. This algorithm enables engineers to find the optimum solutions reliably.

Using the robust optimization algorithm mentioned earlier, in this research a systematic framework for design and synthesis of processes incorporating a multitude of criteria was developed. The methodology incorporates MLMOPSO optimization algorithm which enables handling complexities of process design. A novel decision making methodology also was developed which help decision makers to select a single desired solution after completion of the optimization among many non-dominated optimal solutions.In addition, sustainability metrics developed by other researchers were integrated with the framework to enable evaluation of process alternatives in terms of sustainability concerns.

The application of the proposed framework on DME process retrofit demonstrated the applicability and efficiency of the framework. This framework can be used by engineers for reliable process optimization conveniently without compromising on optimality of solutions due to complexity of process models. The proposed decision making method reduces the cognition load on decision makers significantly and enables them to find a desirable optimal. Thus, this study in the lack of a systematic tool for rigorous process design with a multitude of objectives, contributes significantly in advancement of process system engineering in the field of multi-criteria design.

The application of this research extends well beyond the sustainability considerations to the vast area of optimum design with multi-criteria. The application of this method broadens to any problem involved with selection of the “best” alternative among a choice of multiple process technologies. Application of the methodology could have provided significant reduction of impact on the environment, especially those emerging long-term environmental effects. This research promotes the transition from traditional emission-oriented to impact-oriented industrial pollution prevention. Furthermore, this tool enables engineers to conveniently design optimum processes considering more than one objective not restricted to sustainability concerns.

This research raises the recognition towards sustainability globally, especially for the chemical manufacturing industry. While this research focused on sustainable chemical process design, the application extends well beyond the sustainability considerations any multi-criteria decision making problem. Moreover, the application extends far beyond the chemical industry into other fields of national importance, such as mining, petroleum, energy, pulp, paper, pharmaceutical and other technologies where there is an enormous variety of decision-making scenarios. Therefore the research will benefit the society in the long run through the development of more economic and environmentally viable processes and products.

7.3 Recommended Future Studies

This research has developed all the required foundations and modules for a systematic framework which enables reliable process design with many objectives. However, there are some facets of this research and the modules of the proposed framework which deserve more exploration and improvement in future studies due to limited time and computational facilities in this work.

- Improvement of constraint handling method in MLMOPSO algorithm.

  Handling highly constrained optimization problems deserve more exploration. When optimization problems are highly involved with equality constraints in nonconvex search areas a very effective constraint handling is required. Hybridization of the MINLP deterministic methods and stochastic methaheuristic methods can combine the features of global search and local precise search. This is recommended for handling highly constrained problems more effectively.
- The optimization algorithm should be improved to handle uncertainties in design variables and parameters. This can be done using Monte Carlo sampling method and

utilization of objective functions statistics. Uncertainties are very common in real world design and optimization and the framework should account for design uncertainties.

- Parallel computation can improve time efficiency of the framework which is a concern for very large industrial applications. Several aspects of parallel computation can be incorporated in programming of different modules of the framework. The optimization algorithm has been structured in such a way which enables parallel computation very conveniently.
- The connection between the process simulator and the optimizer for transferring decision variables and calculated parameters can be improved. Depending on the process simulator which is used for the purpose of this framework, some limitations exist which slows transferring data, extracting special simulated parameters from the process simulator and handling warning and errors. These limitations can be solved by improving the accessible objects from the simulator.
- The Sustainability evaluation module can be improved substantially. There are many other aspects of sustainability which should be added to the current sustainability evaluator such as accounting for energy dissipation in the process and type of fuel being used in the process. Also long term aspects of the sustainability and patterns for material usage, recycling and waste disposal outside the battery of process can be taken to the account as well as life cycle assessment.
- Generalize randomized methods such as Greedy randomized adaptive search procedures can be used for selection of a random number of leaders from *archive*.

References

Achour M.H., Haroun A.E., Schult C.J., Gasem K.A.M. (2005). A new method to assess the environmental risk of a chemical process. Chemical Engineering Process 44(8): 901-909.

Adams W. (2006). The Future of Sustainability: Re-thinking environment and development in the twenty-first Century. Report of the IUCN Renowned Thinkers Meeting: 29-31.

Afgan N., Carvalho M., Hovanov N. (2000). Energy system assessment with sustainability indicators. Energy Policy 28(9): 603-612.

Afshinmanesh, Marandi A., Rahimi-Kian A. (2005). A novel binary particle swarm optimization method ssing artificial immune system. Computer as a Tool, EUROCON 2005.

Aguilar-Lasserre A., Pibouleau, L., Azzaro-Pantel, C., Domenech, S. (2009). Enhanced genetic algorithm-based fuzzy multiobjective strategy to multiproduct batch plant design. Applied Soft Computing 9(4): 1321-1330.

Ahmad B.S., Barton P.I. (1995). Solvent recovery targeting for pollution prevention in pharmaceutical and specialty chemical manufacturing. AIChE Symp Ser 90: 98.

AIChE (2000). Center for Waste Reduction Technologies (CWRT). Collaborative Projects Focus Area: Sustainable Ddvelopment; New York, American Institute of Chemical Engineers.

AIChE (2008). Sustainability Index: Strategic commitment to sustainability. Chemical Engineering Progress 104(2): 18-18.

AIChE (1994). Dow's fire and explosion index hazard classification guide. New York, American Institute of Chemical Engineers.

AlRashidi M.R., EL-Naggar KM. (2010). Long term electric load forecasting based on particle swarm optimization. Applied Energy 87(1): 320-6.

Anastas P., Warner J. (1998). Green Chemistry: theory and practice. Oxford University Press: New York:30.

Anderson J., Battikha A., Elizabeth B., Erica C., Ray D., Joel G., Housein J., Wei H., Tracy, S.L Kwang H., Joao M.-H., Steve M., Kelley R., Bill S. (2001). Check your success- A guide to developing community indicators. Retrieved April26 (2007) from http://www.uap.vt.edu/checkyoursuccess/indicators.html.

Angantyr A., Andersson J., Aidanpaa J.O. (2003). Constrained optimization based on a multiobjective evolutionary algorithm. Evolutionary Computation Congress.

Angira R., Babu B.V. (2006). Optimization of process synthesis and design Problems: A modified differential evolution approach. Chemical Engineering Science 61(14): 4707-4721.

Arcoumanis C., Bae C., Crookes R., Kinoshita E. (2008). The potential of Di-Methyl Ether (DME) as an alternative fuel for compression-ignition engines: A Review. Fuel 87(7): 1014-1030.

Atlee J., Kirchain R. (2006). Operational sustainability metrics assessing metric effectiveness in the context of electronics-recycling systems. Environmental Science, Technology 40: 4506-4513.

Azapagic A., Perdan S., Clift R. (2004) eds. Sustainability development in practice: case studies for engineers and scientist. Process Design for Sustainability: The case of Vinyl Chloride Monomer., John Wiley & Sons, Ltd.

Babu B.V., Angira R. (2002). A differential evolution approach for global optimization of MINLP problems. Proceeding of 4th Asia Pacific Conference on Simulated Evolution and Learning. Singapore. November 2002 18-22. Vol. 2: 880-884.

Babu B.V., Chakole Pallavi G., Syed Mubeen J.H. (2005). Multiobjective differential evolution (MODE) for optimization of adiabatic styrene reactor.Chemical Engineering Science 60(17): 4822-4837.

Bhattacharya I., Samanta S. (2010), Parameter selection and performance study in particle swarm optimization. AIP Conference Proceedings, 2010. 1298(1): 564-570.

Bailie R., Whiting J., Shaelwitz J. (2009). Analysis, synthesis and design of chemical processes. Upper Saddle River, Prentice Hall PTR.

Bakshi B. (2000). A thermodynamic framework for ecologically conscious process systems engineering. Computers and Chemical Engineering (24): 1767-1773.

Bartz-Beielstein T., Limbourg P., Mehnen J., Schmitt K., Parsopoulos K.E., Vrahatis M.N., (2003). Particle swarm optimizers for Pareto Optimization with Enhanced Archiving Techniques in Proc. IEEE Congr. Evol. Comput., Dec 2003 Vol 3: 1780-1787.

Beloff B. (2009). Sustainability – What Does it Mean in Practice?. In the International Conference on the Foundations of Computer-Aided Process Design. Breckenridge.

Beloff B., Lines M., Tanzil D., eds (2005). Transforming sustainability strategy into action. T. John Wiley and Sons: Hoboken, New Jersey.

Beloff B., J. Schwarz, E. Beaver (July 2002). Use sustainability metrics to guide decision-making. Chemical Engineering Progress: 58-63.

Beveridge G. S. G., Schechter R. S. (1970). Optimization: theory and practice. McGraw Hill, New York.

Bard J.F. (1998). Practical bilevel optimization; Nonconvex optimization and its applications. Kluwer Academic Publishers.

Bhaskar V., Gupta S.K., Ray A.K. (2000). Applications of multi-objective optimization in chemical engineering. Reviews in Chemical Engineering (16): 1-54.

Biegler L.T., Grosmann I.E., Westerberg A.W. (1998). Systematic methods of chemical crocess design.

Biegler L.T., Grossmann I.E., Westerberg A.W. (1997). Systematic methods of chemical process design. Prentice Hall, New Jersey.

Binh T.T, Korn U. (1997). MOBES: A multi-objective evolution strategy for constrained optimization problems. In The Third International Conference on Genetic Algorithms. Brno, Czech Republic.

Borwein JM., Lewis AS. (2006). Convex analysis and nonlinear optimization: theory and examples. Springer.

Blum C., Roli A. (2003). Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Computing Surveys 35(3): 268-308.

Brundtland G. (1987). Our common future: The world commission on environment and development. Oxford University Press Oxford.

Butner R. S. (1999). A heuristic design advisor for incorporating pollution prevention concepts in Chemical process design; Clean Products and Processes (1): 164-169.

Cabezas H., Bare J., Mallick S. (1999). Pollution prevention with chemical process simulators: The generalized waste reduction (WAR) algorithm; Full Version. Computers and Chemical Engineering 23(4-5): 623-634.

Cai YP., Huang GH., Yang ZF., Lin, QG. (2009). Tan Community-scale renewable energy systems planning under uncertainty; an interval chance-constrained programming approach. Renewable and Sustainable Energy Reviews 13(4):721-35.

Cardoso M. F., Salcedo R. L., Feyo de Azevedo S., Barbosa D. (1997). A simulated annealing approach to the solution of minlp problems.Computers and Chemical Engineering21(12): 1349-1364.
Carvalho A. (2009). Design of sustainable chemical processes: Systematic retrofit analysis generation and evaluation of alternatives(Dissertation). Tecnica de Lisbua University Institute of Chemical Engineers.
Carvalho A., Gani R., Matos H. (2008). Design of sustainable chemical processes: Systematic retrofit analysis generation and evaluation of alternatives. Process Safety and Environmental Protection 86(5): 328-346.
Card S. (2005). The pollution information site. Retrieved May 1, 2010: http://scorecard.org/health-effects/.
Chankong V., Haimes Y. Y. (1983). Multi-objective decision making theory and methodology. Elsevier Science Publishing, New York.
Charnes A., Cooper W. W. (1977). Goal programming and multiple objective optimization; Part I. European Journal of Operational Research (1): 39-54.
Chen D., Gaofeng W., Zhenyi C . (2008). The inertia weight self-adapting in PSO. Intelligent Control and Automation,WCICA 2008.
Chen H., Wen Y., Waterand M.D., Shonnard D.R. (2002). Design guidance for chemical processes using environmental and economic assessments. Ind Eng Chem Res 41: 4503-4513.
Ciric A.R., Gu D.Y. (1994). Synthesis of nonequilibrium reactive distillation processby MINLP optimization. Aiche Journal 40(9):1479-1487.
Clark P.A., Westerberg A.W. (1983). Optimization for design problems having more than one objective. Computers and Chemical Engineering 7 (4): 259-278.
Clerc M. (1999). The swarm and the queen: towards a deterministic and adaptive particle swarm optimization. Evolutionary Computation. CEC 99. Proceedings of the 1999 Congress on.
Coello Coello, Pulido G. T.,Lechuga, M. S. (2004). Handling multiple objectives with particle swarm optimization.IEEE Transactions on Evolutionary Computation8(3): 256-279.
Coello Coello (2002). Theoretical and Numerical Constraint Handling Techniques Used with Evolutionary Algorithms: A Survey of the State of the Art. Computer Methods in Applied Mechanics and Engineering, 191, 1245–1287.
Coello Coello, Lechuga M.S. (2002). MOPSO: a proposal for multiple objective particle swarm optimization. Proceedings of the Evolutionary Computation 2002. CEC '02. Proceedings of the 2002 Congress – Vol. 02, IEEE Computer Society: 1051-1056.
Coello Coello (2000a). An Updated Survey of GA-Based Multiobjective Optimization Techniques. ACM Computing Surveys, 32 (2), 109-143.
Coello Coello (2000b). Handling Preferences in Evolutionary Multiobjective Optimization: A Survey, In Congress on Evolutionary Computation, Volume 1, IEEE Service Center.
Cohon J. L. (1978). Multi-objective programming and planning. Academic press, New York.
Collet P., J.-P. Rennard (2007). Stochastic optimization algorithms. HAL - CCSd – CNRS.
Connolly D., Lund H., Mathiesen B.V., Leahy M. (2010). A review of computer tools for analyzing the integration of renewable energy into various energy systems. Applied Energy 87(4):1059-82.
Constable D., Curzons D., Cunningham V. (2002). Metrics to green' chemistry - Which are the Best? Green Chemistry 4(6): 521-527.
Costa L., Oliveira P. (2001). Evolutionary algorithms approach to the solution of mixed integer non-linear programming problems. Computers and Chemical Engineering25(2-3): 257-266.
Crowl D., Louvar J. (1989). Chemical process safety: Fundamentals with applications. Upper Saddle River, Prentice Hall PTR.

Dantus M. M., High K.A. (1996). Economic valuation for retrofit of chemical processes through waste minimization and process integration. Industrial and Engineering Chemistry. Vol.35 (12): 4566-4578.
Dantus M. (1999). Methodology for the design of economical and environmental friendly processes: An uncertainty approach (Dissertation). 1999, Oklahoma: Stillwater.
Darton R.(2003). Scenarios and metrics as guides to a sustainable future - The case of energy supply. Process Safety and Environmental Protection 81(B5): 295-302.
Das I., Dennis J.E. (1997). A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multi-criteria optimization problems. Structural Optimization 14 (1): 63-69.
Das S., Abraham A., Amit K.(2008). Particle swarm optimization and differential evolution algorithms: technical analysis, applications and hybridization perspectives advances of computational intelligence in industrial systems.
Dasgupta D., Michalewicz Z. (1997a). Evolutionary algorithms in engineering applications. Springer, New York.
Dasgupta D., Michalewicz Z. (1997b). Evolutionary algorithms: An overview. In Evolutionary Algorithms in Egineering Applications. Dasgupta D., Z. Michalewicz, eds., Springer, Berlin.
da Costa J., Pagan J. (2006). Sustainability metrics for coal power generation in Australia. Process Safety and Environmental Protection 84(B2): 143-149.
Deb K., Pratap A., Agarwal S., Meyarivan T. (2002). A fast and elitist multi-objective genetic algorithm: NSGA-II. IEEE Transaction on Evolutionary Computation (6):182-197. (Code available from http://www.iitk.ac.in/kangal/code/nsga2/nsga2-v1.1.tar.)
Deb K. (2000). An efficient constraint handling method for genetic algorithms. Computational Methods in Applied Mechanical Engineering (186):311-338.
Deb K.(2001). Multi-objective optimization using evolutionary algorithms. Wiley, New York.
Deb K., Pratap A., Meyarivan T. (2001). Constrained test problems for multi-objective evolutionary optimization. In Proceedings of the First International Conference on Evolutionary Multi-criteria Optimization.
Deb K. L., Thiele L., Laumanns M., Zitzler E. (2002). Scalable multi-objective optimization test problems. Evolutionary Computation. CEC '02. Proceedings of the 2002 Congress .
Dennis J.E., Schnabel R.B. (1996). Numerical methods for unconstrained optimization and nonlinear equations. Society of Industrial and Applied Mathematics (SIAM), Philadelphia.
Diwekar U. M. (2003). Introduction to Applied Optimization. Kluwer Academic, Norwell, Mass.
Diwekar U.M., I.E. Grossmann, E.S. Rubin (1992). An MINLP process synthesizer for a sequential modular simulator. Industrial and Engineering Chemistry Research 31(1): 313-322.
Diwekar U.M., Rubin E.S. (1993). Efficient handling of the implicit constraints problem for the ASPEN MINLP synthesizer. Industrial and Engineering Chemistry Research 32(9): 2006-2011.
Diwekar U.M., Kalagnanam J.R. (1997). Efficient sampling technique for optimization under uncertainty. AIChE Journal 43(2): 440-447.
Dorigo M., Maniezzo V., Colorni A. (1996). Ant system: optimization by a colony of cooperating agents. IEEE Trans SystMan Cybern 26(1):29-41.
Dorigo M., Gambardella LM. (1997). Ant colonies for the traveling salesman problem. Biosystems, Elsevier Sci; 43(2):73–81.
Douglas J.M. (1988). Conceptual design of chemical processes. McGraw-Hill, New York.
Edgar T. F., Himmelblau D. M., Lasdon L. S. (2001). Optimization of chemical processes, Second Edition, McGraw-Hill, New York.

Elbeltagi E., Tarek H., Donald G.(2005). Comparison among five evolutionary-based optimization algorithms. Advanced Engineering Informatics19(1): 43-53.
Ehrgott M. (2005). Multi-objective Programming. In Multiple Criteria Decision Analysis: State of the Art Survey, eds, J. Figueira, S. Greco, and M. Ehrgott, Springer, New York.
Elhossini A., Areibi S., Robert D. (2010). Strength pareto particle swarm optimization and hybrid EA-PSO for multi-objective optimization. Evolutionary computation18(1): 127-156.
Fisher W. R., Doherty M. F., Douglas J. M. (1987). Screening of process retrofit alternatives. Ind Eng Chem Res Vol.26(11).
Fieldsend J.E., Singh A.S. (2002). A multi-objective algorithm based swarm optimization, an efficient data structure and turbulence. U.K. workshop on Computational Intelligence:34-44.
Floudas C.A., Akrotirianakis I.G., Caratzoulas S., Meyer C.A., Kallrath J. (2005). Global optimization in the 21st century: advances and challenges. Comput Chem Eng 29(6): 1185-1202
Floudas C. A. (1995). Nonlinear mixed-integer optimization: fundamentals and applications. Oxford University Press, New York.
Floudas C. A. (1999). Deterministic global optimization: theory, methods and applications. Kluwer Academic, Boston.
Floudas C. A., Aggarwal A., Ciric A. R. (1989). Global optimum search for nonconvex NLP and MINLP problems. Computers and Chemical Engineering13(10): 1117-1132.
Fonseca C.M., Fleming P.J. (1995). An overview of evolutionary algorithms in multiobjective optimization. Evol. Comput., 1995. 3(1): 1-16.
Fogel L.J., Owens, A.J.,Walsh, M.J. (1966). Artificial Intelligence through Simulated Evolution. Wiley.
Fu Y., Diwekar M.U., Young D., Cabezas H. (2000). Process design for the environment: a multi-objective framework under uncertainty. Clean products process (2): 92-107.
Goel T., Stander N. (2010 a). A non-dominance-based online stopping criterion for multi-objective evolutionary algorithms. International Journal for Numerical Methods in Engineering 84(Compendex): 661-684.
Goel T., Stander N. (2010 b). A Study on the Convergence of Multiobjective Evolutionary Algorithms. the 13th AIAA/ISSMO
Goh C.K., Goh C.K., Tan K.C., Liu D.S., Chiam S.C. (2010). A competitive and cooperative co-evolutionary approach to multi-objective particle swarm optimization algorithm design. European Journal of Operational Research 202(1): 42-54.
Goldberg D.E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley.
Goyal A., Diwekar M.U. (2001). Multi-objective processing synthesis for designing cost effective environment friendly processes. Proceedings of the Conference on CD of World congress of Chemical Engineering, Australia.
Grossmann I., Guillen-Gosalbez G. (2009). Scope for the application of mathematical programming techniques in the synthesis and planning of sustainable processes. in Proceedings of FOCAPD-: Francis Taylor.
Grossmann I. E., Biegler L.T. (2004). Part II: Future perspective on optimisation. Computers and Chemical Engineering 28:1193.
Grossmann I. E., Sargent R.W.H. (1979). Optimal design of multipurpose chemical plants. Industrial and Engineering Chemistry Process Design and Development18: 343-348.
Gunasekera M.Y. , Edwards D.W. (2003) Estimating the environmental impact of catastrophic chemical releases to the atmosphere an index method for ranking alternative chemical process routes. Process Safe Environ Protect 81(6): 463-474.

Grossmann, I.E., Kravanja Z. (1995). Mixed-Integer Nonlinear Programming Techniques for Process Systems Engineering. Computers & Chemical Engineering. 19(Supplement 1): 189-204.
Grossmann, I.E., Daichendt M.M. (1996) New trends in optimization-based approaches to process synthesis. Computers & Chemical Engineering, 1996. 20(6–7): p. 665-683.
Gundersen T., Grossmann I. E. (1990). Improved optimization strategies for automated heat exchanger network synthesis through physical insights. Computers and Chemical Engineering, Vol. 14 (9): 925-944.
Guthrie K. M. (march 24, 1996). Capital cost estimating. Chemical Engineering 76 (6): 114
Gutowski T., Sekulic D., Bakshi B. (2009). Preliminary thoughts on the application of thermodynamics to the development of sustainability criteria. in IEEE International Symposium on Sustainable Systems and Technology.Tempe, Arizona, USA.
Hadj-Alouane A.B., Bean J.C. (1997). A genetic algorithm for the multiple-choice integer program. Operations Research (45): 92-101.
Haimes Y. Y., Tarvainen K., Shima T. , Thadathil J. (1990). Hierarchical multi-objective analysis of large-scale systems. Hemisphere Publishing, New York.
Hau J.L., Yi H.S., Bakshi B.R. (2007). Enhancing life cycle inventories via reconciliation with the laws of thermodynamics. Ind Ecol 11(4): 1-21.
Heikkila A. (1999). Inherent safety in process plant design. An Index-Based Approach, in Department of Chemical Technology. Helsinki University of Technology, Espoo: 132.
Himmelblau D. M. (1972). Applied nonlinear programming. McGraw-Hill, New York.
Holland J.H. (1975). Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, Michigan.
Hock W., Schittkowski K. (1981). Test examples for nonlinear programming codes. Lecture Notes in Economics and Mathematical Systems. Berlin, Germany. Spring-Verlag : 187.
Ho S.L., Shiyou Y., Guangzheng N., Lo E. W. C., Wong H. C. (2005). A particle swarm optimization-based method for multiobjective design optimizations. Magnetics, IEEE Transactions 41(5): 1756-1759.
Homaifa A., Lai S.H.-V., Qi X. (1994). Constrained optimization via genetic algorithms. Simulation 62 (4): 242-254.
Horstman D., Abata D., Keith J., Oberto L. (2005). Feasibility study of an on-board natural gas to dimethyl ether reactor for dimethyl ether preinjection and enhanced ignition. Journal of Engineering for Gas Turbines and Power 127(4): 909-917.
Horst R., Tuy H. (2003). Global optimization. Deterministic Approaches. Springer.
Horst R., Pardalos P.M., Thoai N.V. (2000). Introduction to global optimization. Nonconvex Optimization and its Applications, Kluwer Academic Publishers.
Hu X. , Eberhart R. (2002). Multiobjective optimization using dynamic neighborhood particle swarm optimization. Proceedings of the Evolutionary Computation. CEC '02. Proceedings of the Congress – Vol. 02, IEEE Computer Society: 1677-1681.
Huetinga R., Reijnders L. (2004). Broad sustainability contra sustainability; The proper construction of
sustainability indicators. Ecological Economics Vol (50): 249- 260.
Hwang C.L., Masud A. S. M. (1979). Multiple -objective decision making -methods and applications: A state-of-the-art survey. Lecture Notes in Economics and Mathematical Systems, Berlin. Springer-Verlag.
ICCA (2002). On the road to sustainability. A Summary of ICCA Chemical Sector Report to UNEP: http://www.icca-at-wssd.org/publications.html.
IChemE Metrics (2002). The sustainability metrics: sustainable development progress metrics recommended for use in the process industry. Retrieved March (2010) : http://www.icheme.org/.

Inamdar S.V., Gupta A.K., Saraf D.N. (2004). Multi-objective optimization of an industrial crude distillation unit using the elitist non-dominated sorting genetic algorithm. Chemical Engineering Res Des 82(A5): 611-623.

International Agency for Research on Cancer (March 28, 2009). Overall evaluations of carcinogenicity to humans. Retrieved November 2009: http://monographs.iarc.fr/ENG/Classification/crthgr02a.php.

Ivakpour J., Kasiri N. (2009). Synthesis of distillation column sequences for nonsharp separations. Industrial and Engineering Chemistry Research 48(18): 8635-8649.

Jensen N., Coll N., Gani R. (2003). An integrated computer-aided System for generation and evaluation of sustainable process alternatives. Clean Technology and Environmental Policy 5: 209-225.

Jingjun Zhang Y.S., Gao R., Dong Y.(2009). An improved multi-objective adaptive niche genetic algorithm based on pareto front. IEEE International Advance Computing Conference, IACC (2009).

Jin X. (2005). Approaching sustainability in engineering design with multiple criteria decision making(Dissertation). Oklahoma State University: Stillwater.

Joines J., Houck C. (1994). On the use of non-stationary penalty functions to solve nonlinear constrained optimization problems with GAs. In Proceedings of the First IEEE Conference on Evolutionary Computation. Fogel D., ed., IEEE Press.

Kahraman C., Kaya I., Cebi S. (2009). Acomparative analysis for multi-attribute selection among renewable energy alternatives using fuzzy axiomatic design and fuzzy analytic hierarchy process. Energy (34) 1603-1616.

Kamath R.S., Grossmann I.E., Biegler L.T. (2010). Aggregate Models Based on Improved Group Methods for Simulation and Optimization of Distillation Systems. Computers & Chemical Engineering, 2010. 34(8): 1312-1319.

Karuppiah R., Grossmann I.E. (2008a). Global optimization of multi-scenario mixed integer nonlinear programming models arising in the synthesis of integrated water networks under uncertainty. Comput Chem Eng (32): 145-160.

Karuppiah R., Grossmann I.E. (2008b). A lagrangean based branch-and-cut algorithm for global optimization of nonconvex mixed-integer nonlinear programs with decomposable structures. Glob Optim 41(2): 163-186.

Kazarlis S., Petridis V. (1998). Varying Fitness Functions in Genetic Algorithms: Studying the Rate of Increase of the Dynamic Penalty Terms. In Parallel Problem Solving from Nature V, A.E. Eiben, T. Back, M. Schoenauer, H.-P. Schwefel, ed., Springer- Verlag.

Kennedy J., Eberhart R. C. (1995). Particle swarm optimization proceedings of the IEEE international conference on neural networks: 39-43.

Kennedy J., Eberhart R. C. (1997). A discrete binary version of the particle swarm algorithm. Systems, Man and Cybernetic. Computational Cybernetics and Simulation. IEEE International Conference.

Khanesar M. A., Teshnehlab M., Shoorehdeli M. A. (2007). A novel binary particle swarm optimization. Control and Automation. MED '07, Mediterranean Conference.

Knowles J. D., Corne D. W. (2000). Approximating the nondominated front using the pareto archived evolution strategy. Evolutionary Computation 8(2): 149-172.

Kocis G. R., Grossmann I. E. (1987). Relaxation strategy for the structural optimization of process flow sheets. Industrial and Engineering Chemistry Research 26(9): 1869-1880.

Kocis G. R., Grossmann I. E. (1988). Global optimization of nonconvex mixed-integer nonlinear programming (MINLP) problems in process synthesis.Industrial and Engineering Chemistry Research 27(8): 1407-1421.

Koziet S., Michalewicz Z. (1999). Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. Evolutionary Computation7(1): 19.

Krotscheck C., Narodoslawsky M. (1996). The sustainable process index a new dimension in ecological evaluation. Ecological Engineering 6(4): 241-258.
Krajnc D., Glavič P. (2003). Indicators of sustainable production. Clean Technologies and Environmental Policy 5(3): 279-288.
Lapidus L., Luus R. (1967). Optimal control in engineering processes. Blaisdell, Waltham, Mass.
Le Riche R., C. Knopf-Lenoir, R.T. Haftka (1995). A segregated genetic algorithm for constrained structural optimization. In Proceedings of the Sixth International Conference on Genetic Algorithms.
Leboreiro J., Acevedo J. (2004). Processes synthesis and design of distillation sequences using modular simulators: a genetic algorithm framework. Computers and Chemical Engineering28(8): 1223-1236.
Lee S., Grossmann I. E. (2001). A global optimization algorithm for nonconvex generalized disjunctive programming and applications to process systems. Computers and Chemical Engineering 25(11-12): 1675-1697.
Leong W.F. , Yen G.G. (2008). PSO-bBased multiobjective optimization with dynamic population size and adaptive local archives. IEEE Transactions on Systems, Man and Cybernetics: Part B 38(5): 1270-1293.
Liao C., Tseng C., Pin L. (2007). A discrete version of particle swarm optimization for flowshop scheduling problems. Computers and Operations Research 34(10): 3099-3111.
Li C., Zhanga X., Zhanga S., Suzukib K. (2009). Environmentally conscious design of chemical processes and products: Multi-optimization method. Chemical Engineering Research and Design Vol(87): 233-243.
Li X., Kraslawski A. (2004). Conceptual process synthesis: past and current trends. Chemical Engineering and Processing: Process Intensification 43(5): 583-594.
Li X.N., Rong B.G., Kraslawski A. (2003). A conflict-based approach for process synthesis with waste minimization. ESCAPE-13, Finland.
Li Chunshan, Zhang X., Zhang S., Suzuki K. (2009). Environmentally conscious design of chemical processes and products: Multi-optimization method. Chem Eng Res and Des 87(2): 233-243.
Lili D., Chaogai X., Guohua L. (2010). An improved immune genetic algorithm based on niche algorithm and Its application. 2nd International Symposium on Information Engineering and Electronic Commerce (IEEC). Piscataway, NJ, USA; Ternopil, Ukraine. IEEE Ternopil Nat. Econ. Univ.
Lou H.H., Kulkarni M.A., Singh A., Hopper J.R. (2004). Sustainability assessment for industrial systems. Ind Eng Chem Res 43(15): 4233-4242.
Lou H.H., Singh A. (2006). Hierarchical pareto optimization for the sustainable development of industrial ecosystems. Ind Eng Chem Res 45: 3265-3279.
Lowery R.P., Mcconville B., Yocum F.H., Hendon S.R. (1993). Closed-loop real time optimization of two bisphenol-A plants. Paper presented at the National AICHE Meeting, Houston, TX. Eshelman L.J., ed., Morgan Kaufmann Publishers.
Luo X., Wen Q.-Y., Fieg G. (2009). A hybrid genetic algorithm for synthesis of heat exchanger networks. Computers and Chemical Engineering 33(6): 1169-1181.
Lu H., Chen W. (2008). Self-adaptive velocity particle swarm optimization for solving constrained optimization problems. Journal of Global Optimization 41(3): 427-445.
Luus R. (2000). Iterative dynamic programming. Chapman and Hall, Boca Raton.
Marler R.T., Arora J.S. (2004). Survey of multi-objective optimization methods for engineering. Structural Multidisciplinary Optimization 26: 369-395.
Martins A., Mata T., Costa C., Sikdar S. (2007). Framework for sustainability metrics. Industrial and Engineering Chemistry Research 46(10): 2962-2973.

McPhee J., Yeh W. (2004). Multi-objective optimization for sustainable groundwater management in semiarid regions. Journal of Water Resources Planning and Management-Asce 130(6): 490-497.
Merzougui A., Hasseine A., Kabouche A., Korichi M. (2011). LLE for the extraction of alcohol from aqueous solutions with diethyl ether and dichloromethane at 293.15 K, parameter estimation using a hybrid genetic based approach. Fluid Phase Equilibria 309(2): 161-167.
Mezura-Montes E., Coello Coello C. A., Senior member, IEEE. (2005). A simple multimembered evolution strategy to solve constrained optimization problems. IEEE Transactions on Evolutionary Computation 9(1):1-17.
Michalewicz Z. (1995a). A survey of constraint handling techniques in evolutionary computation methods. In Proceedings of the 4th Annual Conference on Evolutionary Programming: 135-155, Cambridge, MA, MIT Press.
Michalewicz Z., Schoenauer M. (1996). Evolutionary algorithms for constrained parameter optimization problems. Evolutionary Computation 4 (1): 1-32.
Michalewicz Z. (1995b). Genetic algorithms, numerical optimization, and constraints. In Proceedings of the 6th International Conference on Genetic Algorithms, Eshelman L.J., ed., Morgan Kaufmann Publishers.
Miettinen K. (1999). Nonlinear multi-objective optimization, Kluwer Academic Publishers, Boston.
Mitra K., Deb K., Gupta S.K. (1998). Multi-objective dynamic optimization of an industrial nylon-6 semi-batch reactor using genetic algorithm, J. Appl. Polym. Sci. 69 69–87.
Mostaghim S., Schmeck H. (2008). Distance based ranking in many-objective particle swarm optimization. Parallel Problem Solving from Nature – PPSN X. Rudolph G., T. Jansen, S. Lucas, C. Poloni, N. Beume, Springer, Berlin / Heidelberg. 5199: 753-762.
Mostaghim S., Teich J. (2003). Strategies for finding good local guides in multi-objective particle swarm optimization (MOPSO). Swarm Intelligence Symposium. SIS '03, Proceedings of the IEEE.
Nocedal J., Wright S.J. (1999). Numerical optimization. Springer, New York.
Ogawa T., N. Inouem, T. Shikada, O. Inokoshi, O. Y. (2004). Direct dimethyl ether (DME) synthesis from natural gas. Elsevier Science Bv.
Othmana M., Repke J., Wozny G., Huang Y. (2009). A modular approach to sustainability assessment and decision support in chemical process design. Industrial and Engineering Chemistry Research.
Pahl G., Beitz W., Feldhusen J., Grote K.H. (1996). Engineering design: a systematic approach. Springer-Verlag, Berlin/ Heidelberg /New York.
Pulido G., Coello Coello C. (2004), Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer genetic and evolutionary computation. GECCO 2004. Springer Berlin / Heidelberg. p. 225-237
Parsopoulos K.E., Vrahatis M.N. (2002). Particle swarm optimization method in multiobjective problems. in proceedings of the 2002 ACM Symposium on Applied computing., ACM: Madrid, Spain: 603-607.
Pennington D.W. (1997). A pollution prevention tool for continuous chemical processes (P2TCP). PhD Thesis, Hong Kong University of Science and Technology
Piluso C., Huang Y. (2009). Collaborative profitable pollution prevention: An approach for the sustainable development of complex industrial zones under uncertain information. Clean Technologie and Environmental Policy11(3): 307-322.
Ponce-Ortega J. M., Jimenez-Gutierrez A., Grossmann I.E. (2008). Simultaneous retrofit and heat integration of chemical processes. Ind Eng Chem Res Vol. 47:5512-5528.

Powell D., M.M. Skolnick (1993). Using genetic algorithms in engineering design optimization with non-linear constraints. In Proceedings of the Fifth International Conference on Genetic Algorithms, eds. Forrest S., Morgan Kaufmann Publishers.

Rajesh J.K., Gupta S.K., Rangaiah G.P., Ray A.K. (2000). Multiobjective optimization of steam reformer performance using genetic algorithm. Ind Eng Chem Res (39): 706-717.

Ramzan N., Witt W. (2006). Methodology for decision support among conflicting objectives using process simulators. Compute Aided Chemical Engineering 21(Part 1): 415-420.

Ramzan N., Degenkolbe S., Witt W. (2008). Evaluating and improving environmental performance of HC's recovery system: a case study of distillation unit. Chemical Engineering Journal 140(1-3): 201-213.

Rangaiah G.P. (2009). Multi-objective optimization: techniques and applications in chemical engineering, world scientific.

Raquel C. R., Naval Prospero J. C. (2005). An effective use of crowding distance in multiobjective particle swarm optimization. Proceedings of the 2005 Conference on Genetic and Evolutionary Computation. Washington DC, USA, ACM: 257-264.

Ratnaweera A., Halgamuge S. K., Watson H. C. (2004). Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients. IEEE Transactions on Evolutionary Computation 8(3): 240-255.

Ray T., Tai K., Seow K.C. (2001). An evolutionary algorithm for multi-objective optimization. Engineering Optimization 33 (3): 399-424.

Ray W.H., Szekely J. (1973). Process optimization with applications in metallurgy and chemical engineering. Wiley, New York.

Rechenberg I. (1965). Cybernetic Solution Path of an Experimental Problem. Ministry of Aviation, Royal Aircraft Establishment.

Reklaitis G.V., Ravindran A., Ragsdell K.M. (2006). Engineering optimization: Methods and applications. Second Edition, John Wiley, New Jersey.

Rubinov A. (2000). Abstract convexity and global optimization. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers Richardson J.T., M.R. Palmer, G. Liepins, M. Hilliard (1989). Some Guidelines for Genetic Algorithms with Penalty Functions. In Proceedings of the Third International Conference on Genetic Algorithms.

Runarsson T.P., Yao X. (2000). Stochastic ranking for constrained evolutionary optimization. Evolutionary Computation, IEEE Transactions 4(3): 284-294.

Ryoo H.S., Sahinidis N.V. (1995). Global optimization of nonconvex NLPs and MINLPs with applications in process design.Computers and Chemical Engineering19(5): 551-566.

Sakawa M. (1993). Fuzzy sets and interactive multi-objective optimization New York: Plenum Press.

Salcedo R.L. (1992). Solving nonconvex nonlinear programming and mixed-integer nonlinear programming problems with adaptive random search. Industrial and Engineering Chemistry Research31(1): 262-273.

Saling P., Wall C., Wittinger R., Kicherer A. (2002). Eco-efficiency: A tool to demonstrate the sustainability of BASF products. AIChE Meeting, Indianapolis, IN.

Savadkouhi L., Jazayeri S. A., Shahangian N., Tavakoli J. (2010). Performance and combustion characteristics of OM314 diesel engine fueled With DME: A theoretical and experimental analysis. Journal of Engineering for Gas Turbines and Power 132(9): 092801-092806.

Sawaya N. (2006). Reformulations, relaxations and cutting planes for generalized disjunctive programming. PhD Thesis, Pittsburgh, PA, Carnegie Mellon University.

Sawaragi Y., Nakayama H., Tanino T. (1985). Theory of multi-objective optimization. Academic Press, Orlando, Florida.

Schembecker G., Simmrock K.H. (1997). Heuristic-numeric design of separation processes for azeotropic mixtures. Comput Chem Eng Vol. 21, Suppl: S231-S236.

Seager T. (2008). The Sustainability spectrum and the sciences of sustainability. Business Strategy and the Environment: 444-453.
Seager T., Lambert J., Gardner K. (2007). Fostering innovation in contaminated sediments management through multicriteria technology assessment and public participation. Society for Risk Analysis 27(4): 1043-1052.
Seider W., Seader J., Lewin D. (2008). Product and process design principles: Synthesis, analysis and design. New York, John Wiley and Sons.
Semelsberger T.A., Borup R.L., Greene H.L. (2006). Dimethyl ether (DME) as an alternative fuel. Journal of Power Sources 156(2): 497-511.
Sengupta D., Pike R., Hertwig T., Lou H. (2008). Developing and Integrating Sustainable Chemical Processes into Existing Petro-Chemical Plant Complexes. AIChE Annual National Meeting. Philadelphia, Pennsylvania.
Shadiya O., (2010), Social, economic, and environmental metrics for the sustainable optimization of chemical processes(Dissertation).2010, Oklahoma State University: Stillwater.
Sherali H.D., Adams W.P. (1999). A reformulation-linearization technique for solving discrete and coninuous nonconvex problems. Nonconvex Optimization and its Applications, Kluwer Academic Publishers.
Shi Y.H., Eberhart R., BChen Y. (1999). Implementation of evolutionary fuzzy systems. IEEE Transactions on Fuzzy Systems 7(2): 109-119.
Shi Y., Eberhart R.C. (1998). Parameter selection in particle swarm optimization. Evolutionary Programming VII, Springer, Lecture Notes in Computer Science (1447) 591-600.
Shin W.S., Ravindran A. (1991). Interactive multiple objective optimization: survey I – continuous Case. Computers and Operations Research 18(1): 97-114.
Shu X., Zifa L., Jianhua, Z., Qi, Y.(2009). Research on Adaptive Niche Differential Evolution Algorithm for Reactive Power Optimization. in Proceedings 8th International Conference on Advances in Power System Control, Operation and Management (APSCOM 2009). Place of Publication: Stevenage, UK; London, UK. Country of Publication: UK.: IET.
Sierra M., Coello Coello C. (2006). Multi-objective particle swarm optimizers: a survey of the state-of-the-art. International Journal of Computational Intelligence Research, Vol. 2, (3): 287–308.
Sikdar S. (2009). On aggregating multiple indicators into a single metric for sustainability. Clean Technologies and Environmental Policy 11(2): 157-161.
Shokrian M., High K. A. (2012). An efficient multi-objective particle swarm optimization with several global leaders. Manuscript submitted for publication.
Shokrian M., High K. A. (2012). A modified multi-leader particle swarm optimization algorithm for handling NLP/MINLP problems. Manuscript submitted for publication.
Smith A.E., Tate D.M. (1993). Genetic optimization using a penalty function. In Proceedings of the Fifth International Conference on Genetic Algorthms, Forrest S., ed., Morgan Kaufmann Publishers.
Smith R. (1995) Chemical process design, McGraw-Hill, New York.
Soroudi A., Ehsan M ., Zareipour H. (2011). A practical eco-environmental distribution network planning model including fuel cells and non-renewable distributed energy resources. Renewable Energy 36(1):179-88.
Stadler W. (1988). Multi-criteria optimization in engineering and in the sciences. Plenum Press, New York.
Strongin R.G., Sergeyev Y.D. (2000). Global optimization with non-convex constraints; Sequential and parallel algorithms. Nonconvex Optimization and Its Applications, Kluwer Academic Publishers.
Srinivas N., Deb K. (1994). Multi-objective function optimization using non-dominated sorting genetic algorithms. Evolutionary Computation Journal 2 (3): 221-248.

Sun L., Lou H. (2008). A Strategy for multi-objective optimization under uncertainty in chemical process design. Chinese Journal of Chemical Engineering 16(1): 39-42.
Surry P.D., Radcliffe N.J. (1997). The COMOGA method: constrained optimization by multi-objective genetic algorithms. Control and Cybernetics 26 (3).
Tanzil D., Beloff R.B. (2006). Assessing impacts : Overview of sustainability indicators and metrics, environmental quality management: 41-55.
Tarafder A., Rangaiah G.P., Ray A.K. (2007). A study of finding many desirable solutions in multi-objective optimization of chemical processes. Computers and Chemical Engineering (31): 1257-1271.
Tawarmalani M., Sahinidis N.V. (2002). Convexification and global optimization in continuous and mixed-integer nonlinear programming. Theory, Algorithms, Software and Applications. Kluwer Academic, Dordrecht.
Tawarmalani M., Sahinidis N.V. (2005). A polyhedral branch-and-cut approach to global optimization. Math Program103(2):225-249.
The Sustainability Metrics (2002). Sustainable development progress metrics recommended for use in the process industry : http://www.icheme.org/.
Tugnoli A., Santarelli F., Cozzani V. (2008b). An approach to quantitative sustainability assessment in the early stages of process design. Environmental Science and Technology 42(12): 4555-4562.
Turton R., Bailie R., Whiting J., Shaelwitz J. (2009). Analysis, synthesis and design of chemical processes upper saddle river. Prentice Hall PTR.
Tuy H. (1998). Convex analysis and global optimization. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers.
Umeda T., Kuriyama T., Kobayashi S., Ichikawa A. (1980). Interactive solution to multiple criteria problems in chemical process design. Computers and Chemical Engineering (4): 157-165.
U.S Environmental Protection Agency(2006). Green Engineering. [cited 2007 May 31, 2007]: http://www.epa.gov/oppt/greenengineering/pubs/whats_ge.html.
Venkatraman S. (1996). Process Enhancement by Process Simulation and Multiobjective Optimization (M.S. Thesis). Oklahoma State University: Stillwater.
Wang Y., Yang Y. (2009). Particle swarm optimization with preference order ranking for multi-objective optimization. Information Sciences179(12): 1944-1959.
Wu Z., Zhou J. (2007). A self-adaptive particle swarm optimization algorithm with individual coefficients adjustment. Computational Intelligence and Security, International Conference.
Xiaohui H., Eberhart R. (2002). Multiobjective optimization using dynamic neighborhood particle swarm optimization. in Proceedings of the Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600). Piscataway, NJ, USA; Honolulu, HI,USA.: IEEE.
Xu J., Liu Z. (2009). An improved particle swarm optimization algorithm for MINLP problems. Intelligent Systems, GCIS '09. WRI Global Congress.
Yeomans H., Grossmann I.E. (1999). A systematic modeling framework of superstructure optimization in process synthesis. Computers and Chemical Engineering 23(6): 709-731.
Yi H.S., Bakshi B.R. (2007). Rectification of multiscale data with application to life cycle inventories. AIChE 53(4): 876-890.
Yiqing L., Xigang Y., Yongjian L. (2007). An improved PSO algorithm for solving non-convex NLP/MINLP problems with equality constraints. Computers and Chemical Engineering 31(3): 153-162.
Young I.L.L., Floquet P., Joulia X., Kim D.S. (1999). Multi-objective optimization in terms of economics and potential environmental impact for process design and analysis in a chemical process simulator. Ind Eng Chem Res (38): 4729–4741.

Yuan W., Odjo A., Sammons N., Caballero J., Eden M. (2009). Process structure optimization using a hybrid disjunctive-genetic programming approach. in Computer Aided Chemical Engineering, C.A.O.d.N. Rita Maria de Brito Alves, Evaristo Chalbaud Biscaia, Jr., Editors.,Elsevier :669-674.

Zeleny M. (1973). compromise programming in multiple criteria decision making. University of South Carolina Press, Columbia: 261-301.

Zhigljavsky A., Zilinskas A. (2007). Stochastic global optimization. Optimization and Its Applications, Springer.

Zitzler E., Thiele L. (1999). Multiobjective evolutionary algorithms : A comparative case study and the strength Pareto approach. New York, NY, USA Institute of Electrical and Electronics Engineers.

APPENDICES

APPENDIX A: SUSTAINABILITY TABLES

Table 1- Potency Factors for Chemicals that Cause Global Warming (IChemE Metrics, 2002)

Substances	Potency Factor
Carbon dioxide	1
Carbon monoxide	3
Carbon tetrachloride	1400
Chlorodifluoromethane, R22	1700
Chloroform	4
Chloropentafluoroethane, R115	9300
Dichlorodifluoromethane, R12	8500
Dichlorotetrafluoroethane, R114	9300
Difluoroethane	140
Hexafluoroethane	9200
Methane	21
Methylene chloride	9
Nitrogen oxides (NOx)	40
Nitrous oxide	310
Pentafluoroethane, R125	2800
Perfluoromethane	6500
Tetrafluoroethane	1300
Trichloroethane (1,1,1)	110
Trichlorofluoromethane, R11	4000
Trichlorotrifluoroethane, R113	5000
Trifluoroethane, R143a	3800

Table 2- Potency Factors for Chemicals that Cause Stratospheric Ozone Depletion (IChemE Metrics, 2002)

Substances	Potency Factor
Carbon dioxide	1
Carbon monoxide	3
Carbon tetrachloride	1400
Chlorodifluoromethane, R22	1700
Chloroform	4
Chloropentafluoroethane, R115	9300
Dichlorodifluoromethane, R12	8500
Dichlorotetrafluoroethane, R114	9300
Difluoroethane	140
Hexafluoroethane	9200
Methane	21
Methylene chloride	9
Nitrogen oxides (NOx)	40
Nitrous oxide	310
Pentafluoroethane, R125	2800
Perfluoromethane	6500
Tetrafluoroethane	1300
Trichloroethane (1,1,1)	110
Trichlorofluoromethane, R11	4000
Trichlorotrifluoroethane, R113	5000
Trifluoroethane, R143a	3800
Trifluoromethane, R23	11700

Table 3- Potency Factors for Chemicals that Result in the Formation of Photochemical Smog (IChemE Metrics, 2002)

Substances	Potency Factor
1,1-Dichloroethylene	0.232
1,2,3-Trimethylbenzene	1.245

1,2,4- Trimethylbenzene	1.324
1,3,5- Trimethylbenzene	1.299
1-Butene	1.130
1-Pentene	1.040
2,2-Dimethylbutane	0.321
2,3-Dimethylbutane	0.943
2-Butene	0.990
2-Methylbut-1-ene	0.830
2-Methylbut-2-ene	0.770
2-Methylheptane	0.694
2-Methylhexane	0.719
2-Methylnonane	0.657
2-Methyloctane	0.706
2-Methylpentane	0.778
2-Pentene	0.950
3,5-Diethyltoluene	1.195
3,5-Dimethylethylbenzene	1.242
3-Methylbut-1-ene	1.180
3-Methylhexane	0.730
3-Methylpentane	0.661
Acetaldehyde	0.650
Acetic acid	0.156
Acetone	0.182
Acetylene	0.280
Benzaldehyde	-0.056
Benzene	0.334
Butyl glycol	0.629
Butylene	0.703
Butyraldehyde	0.770
Carbon monoxide	0.027
cis 1,2- Dichloroethylene	0.172
Cyclohexane	0.595

Cyclohexanol	0.622
Cyclohexanone	0.529
Diacetone alcohol	0.617
Dimethyl ether	0.263
Ethane	0.140
Ethyl acetate	0.328
Ethyl alcohol	0.446
Ethylbenzene	0.808
Ethylene	1.000
Formaldehyde	0.554
Formic acid	0.003
i-Butane	0.426
i-Butanol	0.591
i-Butyraldehyde	0.855
i-Pentane	0.599
i-Propanol	0.216
i-Propyl acetate	0.291
i-Propylbenzene	0.744
Isoprene	1.180
Methane	0.034
Methyl acetate	0.046
Methyl alcohol	0.205
Methyl chloride	0.035
Methyl cyclohexane	0.732
Methyl- i -butylketone	0.843
Methyl- t -butyl ether	0.268
Methyl chloroform	0.002
Methylene chloride	0.031
Methylethylketone	0.511
m-Ethyltoluene	0.985
m-Xylene	0.080
n-Butane	0.600

Table 4- Potency Factors for Chemicals that Cause *Ecotoxicity* (IChemE Metrics, 2002)

Substances	Potency Factor
1,2-Dichloroethane (EDC)	0.50
Ammonia	0.24
Arsenic	0.20
Benzene	0.17
Cadmium	2.00
Carbon tetrachloride	0.42
Chloride	0.50
Chlorobenzene	1.00
Chloroform	0.42
Chromium	0.33
Copper	1.00
Cyanide	1.00
Formaldehyde	1.00
Hexachlorobenzene	166.67
Hexachlorobutadiene	50.00
Iron	0.005
Lead	0.20
Manganese	0.10
Mercury	16.67
Methylene chloride	0.50
Nickel	0.17
Nitrobenzene	0.25
Nitrophenol	0.50
Tetrachloroethylene (PER)	0.50
Toluene	0.13
Trichloroethylene (TRI)	0.50

Vanadium	0.05
Xylenes	0.17
Zinc	0.13

Table 5- Potency Factors for Chemicals that Cause Aquatic Oxygen Demand (IChemE Metrics, 2002)

Substances	Potency Factor
Acetic acid	1.07
Acetone	2.09
Ammonium nitrate in solution	0.80
Ammonium sulphate in solution	1.00
Chlorotrifluoroethane	0.54
1,2 – Dichloroethane (EDC)	0.81
Ethylene	1.00
Ethylene glycol	1.29
Ferrous ion	0.14
Methanol	1.50
Methyl methacrylate	1.50
Methylene Chloride	0.47
Phenol	2.38
Vinyl chloride	1.28

Table 6- Potency Factors for Chemicals that Cause Atmospheric Acidification (IChemE Metrics, 2002)

Substances	Potency Factor
Ammonia, NH3	1.88
Sulfuric acid mist, H2SO4	0.65
Hydrochloric acid, HCL	0.88
Hydrogen fluoride, HF	1.60

Nitrogen dioxide, NO2	0.70
Sulfur dioxide, SO2	1.00

Table 7- Potency Factors for Chemicals that Cause Eutrophication (IChemE Metrics, 2002)

Substances	Potency Factor
Ammonia	0.33
COD	0.02
Nitrogen	0.42
Nitrogen dioxide, NO2	0.20
Nitrogen oxide, NO	0.13
NOx	0.13
Phosphorus	3.06
PO4 (III-)	1.00

Table 8- Index Score for Heat of Reaction

Mass Enthalpy(H_f) (J/g)	Score
≤ 200	0
<600	2
< 1200	4
< 3000	6
3000	8

Table 9- Index Score for Flammability Index

Flammability Limits (^oC)	Score

Not Flammable	0
Flash Point > 55	2
Flash Point ≤ 55	4
Flash Point < 21	6
Flash point < 0 & boiling point ≤ 35	8

Table 10- Index Score for Explosivity Index

Explosiveness Limit	Score
Not Explosive	0
0-20	2
20-45	4
45-70	6
70-100	8

Table 11- Index Score for Corrosive Index

Material of Construction	Score
Carbon Steel	0
Stainless Steel	2
Better Material Needed	4

Table 12- Index Score for Temperature Index

Temperature (^{o}C)	Score
< 0	2
0-70	0
70-150	2

150-300	4
300-600	6
>600	8

Table 13- Index Score for Pressure Index

Pressure (bar)	Score
0.5 – 5	0
0-0.5 or 5-25	2
20-25	4
50-200	6
200-1000	8

Table 14- Index Score for Equipment Process Safety Index

Type of Equipment	Score
Equipment handling nonflammable, nontoxic materials	0
Heat exchangers, pumps, towers, drums	2
Air coolers, reactors, high hazard pumps	4
Compressors, high hazard reactors	6
Furnaces, fired heaters	8

Table 15- Index Score for Process Safety Structure Index

Process Reliability	Score
Safe	0

Sound Engineering Practice	2
No data	4
Probably Unsafe	6
Minor Accidents	8
Major Accidents	10

Table 16- Index Score for Toxic Exposure Index

Toxic Exposure Limit (ppm)	Score
TLV > 10000	0
TLV ≥ 10000	4
TLV ≤ 1000	8
TLV ≤ 100	12
TLV ≤ 10	16
TLV ≤ 1	20
TLV ≤ 0.1	24
TLV ≤ 0.01	30

Table 17- Index Score for Process Safety

Index Safety Metric	Index Range
Heat of Main Reaction Index	0-8
Heat of Side Reaction Index	0-8
Flammability Index	0-8
Explosiveness Index	0-8
Corrosiveness Index	0-4
Temperature Index	0-8
Pressure Index	0-8
Equipment Safety Index	0-8
Safety Level of Process	0-10

Structure Index	
Toxic exposure Index	0-30
Overall Safety Index	0-100

Table 18- Index Score for Carcinogenic Risk

Type of Carcinogen	Group	Score
Not Carcinogenic	N/A	0
Probably not carcinogenic to humans	4	0.2
Not classifiable as to its carcinogenicity to humans	3	0.4
Possibly carcinogenic	2B	0.6
Probably carcinogenic to humans	2A	0.8
Carcinogenic to humans	1	1

Table 19- Index Score for Health Metrics

Health Metric	Index Range
Carcinogenic Risk	0-1
Developmental Damage	0.6 or 1
Reproductive System Damage	0.6
Circulatory System Damage	0.6
Skeletal System Damage	0.6
Endocrine System Damage	0.6
Gastrointestinal and Liver Damage	0.6
Immune System Damage	0.6
Kidney Damage	0.6
Skeletal System Damage	0.6
Nervous System Damage	0.6

Respiratory System Damage	0.6
Sensory System Damage	0.6

APPENDIX B

Appendix B: MLMOPSO main algorithm MATLAB code

Comments in the following programming code have been reduced to avoid confusion. The commented version of the program is available in the attached MATLAB files.

```
%% Main PSO for MINLP Multiple objective PSO Aug 20th 2011 by MAZDAK
%% SHOKRIAN

%==========this algorithm is for minimization problem with one or more
%optimization onjectives.
clear
for man = 1:1

    clc

    close all

    %==========initialization methods
    Initialization_method = 'Hammesley';

    %===========local best selection methods
    localbeset_selection_method = 'distance_to_Paretofront';

    %============== selection for archive member
    selection_method = 'Hypercube';

    %==================== position updating methods
    updating_method = 'several_leaders';
    %==============Constraints handling


Constraint_method = 'soft';
    %==============Stopping criteria methods

    stopping_meth = 'Con_Rat';

    lag=15; % for stopping criteria
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Algorithmic Paramses
    Nleaders_max = 30;
    leaders_percentage = 0.2;
    Archivesize_max = 100;
    alfa = 1;
    SwarmSize = 100;
    MaxIter = 1000;
    stopping_criteria_count_max = 15;
    %%%%++++++++++++==========initial valued of the parameters which could be made
adaptive
    lag_SIF = 10;
    alfa_o = 0.1;
    C1o = 2.8; C2 = 4 - C1o;
    C1o_descrete = 2.1; C2o_descrete = 4 - C1o_descrete;
    Wo=0.8;
    Woclass=0.4;
    Vbinarymax=6;
    Vbinarymin=-6;
    perturbation_percent=0.0;

    axix_scaler=0.1225;
    calss_var_updatingfrequency = 0.7;

    %%%%%%%variables definition
    [VarHigh VarLow N_variables N_var_class] = variable_bound_reader();
    %======check the problem parameters==========

    if (Nleaders_max == 0)||(Archivesize_max == 0)
        Nleaders_max =1;
        Archivesize_max =1;
    end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%% Initialization Swarm
    n=SwarmSize;L_Archive=0;
    d=N_variables;
    Iter=0
    C_1 = strcmp(Initialization_method,'Hammesley');
    if C_1 == 1
        violant_percent = 1;
        violant_num = 0;
        Samples = hammesley(n,d);
        for ii = 1:SwarmSize
            Particle(ii).Position.continuous = Samples(ii,:).* (VarHigh - VarLow) + VarLow;
            Particle(ii).LocalBestPosition.continuous = Particle(ii).Position.continuous;

            Particle(ii).Position.descrete = round(rand(1,N_var_class)) + 0;
            Particle(ii).LocalBestPosition.descrete = Particle(ii).Position.descrete;

            Particle(ii).Velocity.continuous = (rand(1,N_variables)-0.5).* (VarHigh - VarLow);
```

```
            Particle(ii).Velocity.descrete.zero = (rand(1,N_var_class));
            Particle(ii).Velocity.descrete.one = (rand(1,N_var_class));

            [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(ii),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy);
            Particle(ii).Cost = Cost;
            Particle(ii).constraint_violation =Violation;
            Particle(ii).alphaa = alphaa;
            Particle(ii).LocalBestCost = Particle(ii).Cost;
            Particle(ii).overalviolation =Overalviolation;
            Particle(ii).LocalBest_constraint_violation = Violation;
            Particle(ii).LocalBest_overalviolation = Overalviolation;
            NumOFs =length(Cost);
            if Particle(ii).constraint_violation == 1
                violant_num = violant_num + 1;
            end

        end
        violant_percent = violant_num/SwarmSize;

    end
    C_1 = strcmp(Initialization_method,'Random');
    if C_1 == 1
        violant_percent = 1;

        violant_num = 0;
        for ii = 1:SwarmSize
            Particle(ii).Position.continuous = rand(1,N_variables) .* (VarHigh - VarLow) + VarLow;
            Particle(ii).LocalBestPosition.continuous = Particle(ii).Position.continuous;
            Particle(ii).Velocity.continuous = (rand(1,N_variables)-0.5 ).* (VarHigh - VarLow);

            Particle(ii).Position.descrete = round(rand(1,N_var_class)  + 0);
            Particle(ii).LocalBestPosition.descrete = Particle(ii).Position.descrete;
            Particle(ii).Velocity.descrete.zero = (rand(1,N_var_class) );
            Particle(ii).Velocity.descrete.one = (rand(1,N_var_class) );


            [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(ii),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy);
            Particle(ii).Cost = Cost;
            Particle(ii).constraint_violation =Violation;
            Particle(ii).alphaa = alphaa;
            Particle(ii).LocalBestCost = Particle(ii).Cost;
            Particle(ii).overalviolation =Overalviolation;
            Particle(ii).LocalBest_constraint_violation = Violation;
            Particle(ii).LocalBest_overalviolation = Overalviolation;
            NumOFs =length(Cost);
            if Constraint_method == 'hard'
                while Particle(ii).constraint_violation == 1
                    Particle(ii).Position.continuous = rand(1,N_variables) .* (VarHigh - VarLow) +
VarLow;
```

```
            Particle(ii).LocalBestPosition.continuous = Particle(ii).Position.continuous;
            Particle(ii).Velocity.continuous = (rand(1,N_variables)-0.5 ).* (VarHigh - VarLow);
            Particle(ii).Position.descrete = round(rand(1,N_var_class)  + 0);
            Particle(ii).LocalBestPosition.descrete = Particle(ii).Position.descrete;
            Particle(ii).Velocity.descrete.zero = (rand(1,N_var_class) );
            Particle(ii).Velocity.descrete.one = (rand(1,N_var_class) );

            [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(ii),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy);
            Particle(ii).Cost = Cost;
            Particle(ii).constraint_violation =Violation;
            Particle(ii).alphaa = alphaa;
            Particle(ii).LocalBestCost = Particle(ii).Cost;
            Particle(ii).overalviolation =Overalviolation;
            Particle(ii).LocalBest_constraint_violation = Violation;
            Particle(ii).LocalBest_overalviolation = Overalviolation;
            NumOFs =length(Cost);
          end
        end
        if Particle(ii).constraint_violation == 1
          violant_num = violant_num + 1;
        end


      end
      violant_percent = violant_num/SwarmSize;

    end

    PMax_Viol(ii)= Particle(ii).overalviolation;
    Wtransition(ii) = Wo;
    Wtransition_class(ii) = Woclass;

    t=0;
    Non_dominatedset=[]; Sorted_n_d=[];Archive=[];
    for ii=1:SwarmSize
      flag1 = [];
      for jj=1:SwarmSize
        if any(Particle(ii).Cost < Particle(jj).Cost) || (ii == jj)||all(Particle(ii).Cost ==
Particle(jj).Cost)
          if all(Particle(ii).Cost == Particle(jj).Cost) && all(Particle(ii).Position.continuous ==
Particle(jj).Position.continuous)&& all(Particle(ii).Position.descrete ==
Particle(jj).Position.descrete)&& (ii>jj)
            flag1 = 'dominated';
            break;
          else
            flag1 = 'non-dominated';

            if (Iter>=ceil(MaxIter*Iter_constdominancy))
              if (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation == 0)
                flag1 = 'dominated';
```

```
                    break;
                end
                if (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation ==
1)&&(Particle(ii).overalviolation > Particle(jj).overalviolation)
                    flag1 = 'dominated';
                    break;
                end
            end
        end

    else
        if (Iter>=ceil(MaxIter*Iter_constdominancy))
            if (Particle(ii).constraint_violation == 0)&&(Particle(jj).constraint_violation == 1)
                flag1 = 'non-dominated';

            elseif (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation ==
1)&&(Particle(ii).overalviolation < Particle(jj).overalviolation)

                flag1 = 'non-dominated';

            else
                flag1 = 'dominated';

                break;

            end
        else
            flag1 = 'dominated';
            break;
        end
    end
  end
  C_2 = strcmp(flag1,'non-dominated');
  if (C_2 == 1)
    t=t+1;
    if t < 2
      Non_dominatedset = Particle(ii);
    else
      Non_dominatedset(t) = Particle(ii);
    end
  end
end
for k=1:length(Non_dominatedset)
  Non_dominatedset(k).archive_success=0;
end

L_Non_dominatedset = length(Non_dominatedset);
if L_Non_dominatedset > Archivesize_max
  index_selection = randperm(Archivesize_max);
  Archive = Non_dominatedset(index_selection(1));
  for i = 2:Archivesize_max
```

```
        Archive(i) = Non_dominatedset(index_selection(i));
    end

else
    Archive = Non_dominatedset;

end
L_Archive = length(Archive);
for i=1:(L_Archive)
    Archive(i).archive_success = 1; % just for initiallization
    Archive(i).aggravatedfitness=[];

end
Pre_Archive=Archive;

Num_leaders = min(min( Nleaders_max,L_Archive), ceil(leaders_percentage * SwarmSize));

Leaders = Archive((L_Archive - Num_leaders +1): L_Archive);


%=================================================================
============================================================
  dominated_num_inswarm_eachiterMat = [];L_Non_dominatedTempMat = [];
L_Non_dominatedsetMat = [];
  new_MEAN=0; new_VAR = 0; stopping_criteria_count = 0; check_stop =1; n_Q =
0;n_Q_SIF=0;CR=0;CR_SIF=0;
  Iter=1
  while (Iter <= MaxIter) && (check_stop==1)

    L_leaders = length(Leaders);
    Distance_localBest = [];
    Distance=[];Distance_discrete=[];

    if rand > calss_var_updatingfrequency % this determine if class variables are updated or not
        class_update = true;
    else
        class_update = false;
    end
    for ii = 1:SwarmSize

        %===========Dynamic change of alpha

        Viol_particle = Particle(ii).overalviolation;

        alfa = alfa_o*exp(4.4*CR_SIF^0.6);
        %=================Leadre assignment ============================

        for lj = 1:L_leaders
            range1=(VarHigh - VarLow);
            index1 = find((VarHigh - VarLow)==0);
            for lk = 1: length(index1)
```

```
            range1(index1(lk))= VarHigh(index1(lk));
        end
        Distance(lj)= (sum( ((Particle(ii).Position.continuous-
Leaders(lj).Position.continuous)./(range1)).^2)+sum( (Particle(ii).Position.descrete-
Leaders(lj).Position.descrete).^2)) ^ 0.5;
    end

    [Min_Dis I_dis] = min (Distance);
    S_i = alfa * Min_Dis;

    C_C = strcmp(updating_method , 'oneleader_foreach');
    %==========Dynamic change of the parameters

    Viol_localbest = Particle(ii).LocalBest_overalviolation ;
    Viol_Closestleader = Leaders(I_dis).overalviolation;


    if Viol_particle > 0

        C1 = C1o;
        C2 = 4-C1;
        C1_descrete = C1o_descrete;
        C2_descrete = 4-C1_descrete;


    else
        C1 = C1o*exp(-0.84*CR_SIF);
        C2 = 4-C1;
        C1_descrete = C1o_descrete*exp(-0.15*CR_SIF);
        C2_descrete = 4-C1_descrete;

    end

    W =Wo;
    Wclass = Woclass;
    inertia_term = W * Particle(ii).Velocity.continuous;

    %==========continuous and descrete parts
    Social_term_contin = 0;Social_term_descrete=0;
    if (S_i==0)||(C_C==1)
        Coeff = C2;
        Coeff_descrete = C2_descrete;
        Social_term_contin = Social_term_contin + Coeff* rand * (
Leaders(I_dis).Position.continuous - Particle(ii).Position.continuous);

    else
        Coeff = (C2 / sum(exp(-(Distance/S_i).^2)));


        for Li = 1: L_leaders
            coef_holder(Li) = Coeff* exp(-(Distance(Li)/S_i)^2);
```

```
            Social_term_contin = Social_term_contin + Coeff* exp(-(Distance(Li)/S_i)^2) *
rand * ( Leaders(Li).Position.continuous - Particle(ii).Position.continuous );

        end
        Coeff_descrete = C2_descrete;
    end
    %====================finding new Particle(ii).LocalBestPosition==========
    % I can put all these in a function
    new_Particle_position = 'dominated';
    old_Particle_localbest = 'dominated';
    if any(Particle(ii).Cost < Particle(ii).LocalBestCost)
        new_Particle_position = 'non_dominated';
        if (Iter>=ceil(MaxIter*Iter_constdominancy))
            if (Particle(ii).constraint_violation ==
1)&&(Particle(jj).LocalBest_constraint_violation == 0)
                new_Particle_position = 'dominated';
                old_Particle_localbest = 'non_dominated';
            end

            if (Particle(ii).constraint_violation ==
1)&&(Particle(jj).LocalBest_constraint_violation == 1)&&(Particle(ii).overalviolation >
Particle(jj).LocalBest_overalviolation)
                new_Particle_position = 'dominated';
                old_Particle_localbest = 'non_dominated';
            end
        end
    end
    if any(Particle(ii).LocalBestCost < Particle(ii).Cost)
        old_Particle_localbest = 'non_dominated';
        if (Iter>=ceil(MaxIter*Iter_constdominancy))
            if (Particle(ii).constraint_violation ==
0)&&(Particle(jj).LocalBest_constraint_violation == 1)
                new_Particle_position = 'non_dominated';
                old_Particle_localbest = 'dominated';
            end
            if (Particle(ii).constraint_violation ==
1)&&(Particle(jj).LocalBest_constraint_violation == 1)&&(Particle(ii).overalviolation <
Particle(jj).LocalBest_overalviolation)
                new_Particle_position = 'non_dominated';
                old_Particle_localbest = 'dominated';
            end
        end


    end

    C_3 = strcmp(new_Particle_position , 'non_dominated');
    C_4 = strcmp(old_Particle_localbest,'non_dominated');

    if (C_3 == 1)&& (C_4 == 1)
        C_5 = strcmp(localbeset_selection_method , 'distance_to_Paretofront');
```

```
            if C_5 == 1

               for lj=1:L_leaders
                  Distance_localBest(lj)= (sum( (Particle(ii).LocalBestPosition.continuous -
Leaders(lj).Position.continuous).^2)+ sum( (Particle(ii).LocalBestPosition.descrete -
Leaders(lj).Position.descrete).^2)) ^ 0.5;
               end
               Mindist_pre_Localbest = min(Distance_localBest);
               Mindist_particle_newposition = min(Distance);
               if Mindist_pre_Localbest < Mindist_particle_newposition
                  Particle(ii).LocalBestPosition = Particle(ii).LocalBestPosition;
                  Particle(ii).LocalBestCost= Particle(ii).LocalBestCost;
               elseif Mindist_particle_newposition <= Mindist_pre_Localbest
                  Particle(ii).LocalBestPosition = Particle(ii).Position;
                  Particle(ii).LocalBestCost= Particle(ii).Cost;
               end
            end
            C_5 = strcmp(localbeset_selection_method , 'Num_dominated_particles');
            if  C_5 == 1

               Localbestarray(1).Position = Particle(ii).Position;
               Localbestarray(2).Position = Particle(ii).LocalBestPosition;
               Localbestarray(1).Cost = Particle(ii).Cost;
               Localbestarray(2).Cost = Particle(ii).LocalBestCost;

               for i = 1:2
                  Localbestarray(i).aggravatedfitness =
aggeravated_localbest_fit_func(Localbestarray(i),Particle,SwarmSize);

               end
               sorted_Localbestarray = bublesort_fitness(Localbestarray);
               Particle(ii).LocalBestPosition = sorted_Localbestarray(2).Position;
               Particle(ii).LocalBestCost= sorted_Localbestarray(2).Cost;

            end
         end
         if (C_3 == 1)&& (C_4 == 0)
            Particle(ii).LocalBestPosition = Particle(ii).Position;
            Particle(ii).LocalBestCost= Particle(ii).Cost;
            Particle(ii).LocalBest_constraint_violation =Particle(ii).constraint_violation;
            Particle(ii).LocalBest_overalviolation =Particle(ii).overalviolation;
         end
         if (C_3 == 0)&& (C_4 == 1)
            Particle(ii).LocalBestPosition = Particle(ii).LocalBestPosition;
            Particle(ii).LocalBestCost= Particle(ii).LocalBestCost;
            Particle(ii).LocalBest_constraint_violation =
Particle(ii).LocalBest_constraint_violation;
            Particle(ii).LocalBest_overalviolation = Particle(ii).LocalBest_overalviolation;
         end
         if (C_4 == 0)&& (C_3 == 0)
            Particle(ii).LocalBestPosition = Particle(ii).Position;
```

```
            Particle(ii).LocalBestCost= Particle(ii).Cost;
            Particle(ii).LocalBest_constraint_violation =Particle(ii).constraint_violation;
            Particle(ii).LocalBest_overalviolation =Particle(ii).overalviolation;
        end

        %=============updating new positions=======================
        if Constraint_method == 'hard'
            PreviousPosition = Particle(ii).Position;
            PreviousCost = Particle(ii).Cost;
        end

        rr1 = floor((length(Pre_Archive)- 1)*rand + 1);
        rr2 = floor((length(Pre_Archive)- 1)*rand + 1);
        inertia_term = Wo * abs(Pre_Archive(rr1).Position.continuous -
Pre_Archive(rr2).Position.continuous).*sign(Particle(ii).Velocity.continuous);


        inertia_term = W * Particle(ii).Velocity.continuous;


        Particle(ii).Velocity.continuous = inertia_term + C1 * rand *
(Particle(ii).LocalBestPosition.continuous - Particle(ii).Position.continuous) + ...
            Social_term_contin;

        Particle(ii).Position.continuous = Particle(ii).Position.continuous +
Particle(ii).Velocity.continuous;
        Particle(ii).Position.continuous = min(Particle(ii).Position.continuous , VarHigh);
        Particle(ii).Position.continuous = max(Particle(ii).Position.continuous , VarLow);
        if class_update == true
            for kk= 1:N_var_class
                Social_term_descretezero=0;localzero=0;
                Social_term_descreteone=0;localone=0;
                if Particle(ii).Position.descrete(kk)==0


                    if Leaders(I_dis).Position.descrete(kk)== 0
                        r=rand;
                        Social_term_descretezero = Social_term_descretezero + Coeff_descrete*r;
                        Social_term_descreteone = Social_term_descreteone - Coeff_descrete* r;

                    else
                        r=rand;
                        Social_term_descretezero = Social_term_descretezero - Coeff_descrete*r;
                        Social_term_descreteone = Social_term_descreteone + Coeff_descrete* r;
                    end

                    if Particle(ii).LocalBestPosition.descrete(kk)==0
                        r2=rand;
                        localzero = +C1_descrete*r2;
                        localone = -C1_descrete*r2;
```

```
            else
               r2=rand;
               localzero = -C1_descrete*r2;
               localone = +C1_descrete*r2;

            end
            Particle(ii).Velocity.descrete.zero(kk) = Wclass *
Particle(ii).Velocity.descrete.zero(kk) + localzero + Social_term_descretezero;
            Particle(ii).Velocity.descrete.one(kk) = Wclass *
Particle(ii).Velocity.descrete.one(kk) + localone + Social_term_descreteone;
            sig_arg = Particle(ii).Velocity.descrete.one(kk);
            arg1=min(round(1./(1+exp(-sig_arg))-rand + 0.5),1);
            arg1=max(arg1,0);
            Particle(ii).Position.descrete(kk) = abs(arg1 - Particle(ii).Position.descrete(kk));


         else


            if Leaders(I_dis).Position.descrete(kk)==0
               r=rand;
               Social_term_descretezero = Social_term_descretezero + Coeff_descrete*r;
               Social_term_descreteone = Social_term_descreteone - Coeff_descrete* r;

            else
               r=rand;
               Social_term_descretezero = Social_term_descretezero - Coeff_descrete*r;
               Social_term_descreteone = Social_term_descreteone + Coeff_descrete* r;
            end


            if Particle(ii).LocalBestPosition.descrete(kk)==0
               r2=rand;
               localzero = +C1_descrete*r2;
               localone = -C1_descrete*r2;

            else
               r2=rand;
               localzero = -C1_descrete*r2;
               localone = +C1_descrete*r2;

            end
            Particle(ii).Velocity.descrete.zero(kk) = Wclass *
Particle(ii).Velocity.descrete.zero(kk) + localzero + Social_term_descretezero;
            Particle(ii).Velocity.descrete.one(kk) = Wclass *
Particle(ii).Velocity.descrete.one(kk) + localone + Social_term_descreteone;
            sig_arg = Particle(ii).Velocity.descrete.zero(kk);
            arg1=min(round(1./(1+exp(-sig_arg))-rand + 0.5),1);
            arg1=max(arg1,0);
            Particle(ii).Position.descrete(kk) = abs(arg1 - Particle(ii).Position.descrete(kk));
```

```
            end

        end

    end


    [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(ii),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy);
    Particle(ii).Cost = Cost;
    Particle(ii).constraint_violation =Violation;
    Particle(ii).alphaa = alphaa;
    Particle(ii).overalviolation =Overalviolation;
    if (Constraint_method == 'hard') & (Particle(ii).constraint_violation == 1)
        Particle(ii).Position = PreviousPosition;
        Particle(ii).Cost = PreviousCost;
    end

end

% ****************** perturbation ************************
if (CR >= 0.6)|| (stopping_criteria_count >= stopping_criteria_count_max/8)

    if L_Archive ==1
        Num_perturbed =
min(ceil(rand*SwarmSize),round(perturbation_percent*SwarmSize));
        for a=1:Num_perturbed
            shom = round(rand*SwarmSize);
            if shom ==0
                shom=1;
            end
            if rand > 0.63
                Particle(shom).Position.continuous=Archive(1).Position.continuous  +
rand(1,length(Archive(1).Position.continuous));
            else
                Particle(shom).Position.continuous=Archive(1).Position.continuous  +
rand(1,length(Archive(1).Position.continuous)).*(VarHigh-VarLow);
            end
            Particle(shom).Position.continuous = min(Particle(shom).Position.continuous ,
VarHigh);
            Particle(shom).Position.continuous = max(Particle(shom).Position.continuous ,
VarLow);
            Particle(shom).Position.descrete = round(rand(1,N_var_class));
            Particle(shom).Velocity.descrete.zero = (rand(1,N_var_class)-0.5);
            Particle(shom).Velocity.descrete.one = (rand(1,N_var_class)-0.5);
            [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(shom),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy)
;
            Particle(shom).Cost = Cost;
```

```
                Particle(shom).constraint_violation =Violation;
                Particle(shom).alphaa = alphaa;
                Particle(shom).overalviolation =Overalviolation;
            end


        else
            Num_perturbed =
min(ceil(rand*SwarmSize),round(perturbation_percent*SwarmSize));
            for a=1:Num_perturbed
                shom = round(rand*SwarmSize);
                if shom ==0
                    shom=1;
                end
                if rand > 0.8
                    Mut_ind= randperm(L_Archive);
                    Particle(shom).Position.continuous = Archive(Mut_ind(1)).Position.continuous +
rand(1,length(Archive(1).Position.continuous));
                else
                    Mut_ind= randperm(L_Archive);

                    Particle(shom).Position.continuous = Archive(Mut_ind(1)).Position.continuous  +
rand(1,length(Archive(1).Position.continuous)).*(Archive(Mut_ind(2)).Position.continuous -
Archive(Mut_ind(1)).Position.continuous);
                end
                Particle(shom).Position.continuous = min(Particle(shom).Position.continuous ,
VarHigh);
                Particle(shom).Position.continuous = max(Particle(shom).Position.continuous ,
VarLow);
                Particle(shom).Position.descrete = round(rand(1,N_var_class));
                Particle(shom).Velocity.descrete.zero = (rand(1,N_var_class)-0.5);
                Particle(shom).Velocity.descrete.one = (rand(1,N_var_class)-0.5);

                [Cost Violation alphaa Overalviolation] =
Cost_Fcn(Particle(shom),ProbParams,Iter,MaxIter,L_Archive,Iter_extraOF,Iter_constdominancy)
;
                Particle(shom).Cost = Cost;
                Particle(shom).constraint_violation =Violation;
                Particle(shom).alphaa = alphaa;
                Particle(shom).overalviolation =Overalviolation;
            end
        end

    end
    % ********************

    t=0;
    Non_dominatedset=[]; Final_non_dominated_set=[]; dominated_num_inswarm_eachiter =
0;
    Archive=[];
    for ii=1:SwarmSize
```

```
        flag1=[];

        for jj=1:SwarmSize
           if any(Particle(ii).Cost < Particle(jj).Cost) || (ii == jj)||all(Particle(ii).Cost ==
Particle(jj).Cost)
              if all(Particle(ii).Cost == Particle(jj).Cost) && all(Particle(ii).Position.continuous ==
Particle(jj).Position.continuous)&& all(Particle(ii).Position.descrete ==
Particle(jj).Position.descrete)&&(ii>jj)
                 flag1 = 'dominated';
                 dominated_num_inswarm_eachiter = dominated_num_inswarm_eachiter + 1;
                 break;
              else
                 flag1 = 'non-dominated';
                 if (Iter>=ceil(MaxIter*Iter_constdominancy))
                    if (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation ==
0)
                       flag1 = 'dominated';
                       dominated_num_inswarm_eachiter = dominated_num_inswarm_eachiter + 1;
                       break;
                    end
                    if (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation ==
1)&&(Particle(ii).overalviolation > Particle(jj).overalviolation)
                       flag1 = 'dominated';
                       dominated_num_inswarm_eachiter = dominated_num_inswarm_eachiter + 1;
                       break;
                    end
                 end
              end
           else
              if (Iter>=ceil(MaxIter*Iter_constdominancy))
                 if (Particle(ii).constraint_violation == 0)&&(Particle(jj).constraint_violation == 1)
                    flag1 = 'non-dominated';

                 elseif (Particle(ii).constraint_violation == 1)&&(Particle(jj).constraint_violation
== 1)&&(Particle(ii).overalviolation < Particle(jj).overalviolation)

                    flag1 = 'non-dominated';

                 else
                    flag1 = 'dominated';
                    dominated_num_inswarm_eachiter = dominated_num_inswarm_eachiter + 1;

                    break;

                 end
              else
                 flag1 = 'dominated';
                 dominated_num_inswarm_eachiter = dominated_num_inswarm_eachiter + 1;
                 break;
              end
```

```
            end
        end
        C_6 = strcmp(flag1,'non-dominated');
        if (C_6 == 1)

            t=t+1;
            if t<2
                Non_dominatedset = Particle(ii);
            else
                Non_dominatedset(t) = Particle(ii);
            end
        end
    end
    for k=1:length(Non_dominatedset)
        Non_dominatedset(k).archive_success=0;
        Non_dominatedset(k).aggravatedfitness = [];
    end

    Temp=Non_dominatedset;
    L_Non_dominatedset=length(Non_dominatedset);

    Temp=[Temp Pre_Archive];

    if (Iter == ceil(MaxIter*Iter_constdominancy))&& (Iter_constdominancy > 0)

        for k =1:length(Temp)
            Temp(k).Cost(NumOFs)=0;
            Temp(k).LocalBestCost(NumOFs)=0;
        end
        NumOFs = NumOFs-1;
    end

    T_t=0;
    L_Temp = length(Temp);
    newArchivemember_fromswarm=0;
    if (Iter == ceil(MaxIter*Iter_constdominancy)) || (Iter == ceil(MaxIter*Iter_extraOF))

        for it=1:L_Temp
            T_flag1=[];

            newArchivemember_fromswarm=newArchivemember_fromswarm+1;
            for jt= 1:L_Temp
                if any(Temp(it).Cost < Temp(jt).Cost)|| (it == jt)||all(Temp(it).Cost ==
Temp(jt).Cost)

                    if all(Temp(it).Cost == Temp(jt).Cost) && all(Temp(it).Position.continuous ==
Temp(jt).Position.continuous)&& all(Temp(it).Position.descrete ==
Temp(jt).Position.descrete)&&(it<jt)
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
```

```
                    break;
                else
                    T_flag1 = 'non-dominated';
                end
                if (Iter>=ceil(MaxIter*Iter_constdominancy))
                    if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation == 0)
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                        break;
                    end
                    if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
1)&&(Temp(it).overalviolation > Temp(jt).overalviolation)
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                        break;
                    end
                end

            else
                if (Iter>=ceil(MaxIter*Iter_constdominancy))
                    if (Temp(it).constraint_violation == 0)&&(Temp(jt).constraint_violation == 1)
                        T_flag1 = 'non-dominated';

                    elseif (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
1)&&(Temp(it).overalviolation < Temp(jt).overalviolation)

                        T_flag1 = 'non-dominated';

                    else
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;

                        break;

                    end
                else
                    T_flag1 = 'dominated';
                    newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                    break;
                end
            end
        end

        C_7 = strcmp(T_flag1,'non-dominated');
        if C_7 == 1
            T_t=T_t + 1;
            if T_t<2
                Non_dominatedTemp = Temp(it);
            else
                Non_dominatedTemp(T_t) = Temp(it);
```

```
            end
        end
    end


else
    expi=0;
    for it=1:L_Temp
        T_flag1=[];
        if  it <= L_Non_dominatedset
            newArchivemember_fromswarm=newArchivemember_fromswarm+1;
            for jt= (L_Non_dominatedset + 1):L_Temp
                if any(Temp(it).Cost < Temp(jt).Cost)||all(Temp(it).Cost == Temp(jt).Cost)

                    if all(Temp(it).Cost == Temp(jt).Cost ) && all(Temp(it).Position.continuous ==
Temp(jt).Position.continuous)&&(it<jt)
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;

                        break;
                    else
                        T_flag1 = 'non-dominated';

                    end
                    if (Iter>=ceil(MaxIter*Iter_constdominancy))
                        if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
0)
                            T_flag1 = 'dominated';
                            newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                            break;
                        end
                        if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
1)&&(Temp(it).overalviolation > Temp(jt).overalviolation)
                            T_flag1 = 'dominated';
                            newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                            break;
                        end
                    end


                else
                    if (Iter>=ceil(MaxIter*Iter_constdominancy))
                        if (Temp(it).constraint_violation == 0)&&(Temp(jt).constraint_violation ==
1)
                            T_flag1 = 'non-dominated';

                        elseif (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation
== 1)&&(Temp(it).overalviolation < Temp(jt).overalviolation)

                            T_flag1 = 'non-dominated';
```

```
                    else
                        T_flag1 = 'dominated';
                        newArchivemember_fromswarm=newArchivemember_fromswarm - 1;

                        break;

                    end
                else
                    T_flag1 = 'dominated';
                    newArchivemember_fromswarm=newArchivemember_fromswarm - 1;
                    break;
                end
            end
        end

        C_7 = strcmp(T_flag1,'non-dominated');
        if C_7 == 1
            T_t=T_t + 1;
            if T_t<2
                Non_dominatedTemp = Temp(it);
            else
                Non_dominatedTemp(T_t) = Temp(it);
            end
        end
    end

    if  it > L_Non_dominatedset
        for jt= 1:L_Non_dominatedset
            if any(Temp(it).Cost < Temp(jt).Cost)||all(Temp(it).Cost == Temp(jt).Cost)
                T_flag1 = 'non-dominated';

                if (Iter>=ceil(MaxIter*Iter_constdominancy))
                    if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
0)
                        T_flag1 = 'dominated';
                        break;
                    end
                    if (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation ==
1)&&(Temp(it).overalviolation > Temp(jt).overalviolation)
                        T_flag1 = 'dominated';
                        break;
                    end
                end

                else
                    if (Iter>=ceil(MaxIter*Iter_constdominancy))
                        if (Temp(it).constraint_violation == 0)&&(Temp(jt).constraint_violation ==
1)
                            T_flag1 = 'non-dominated';
```

```
                    elseif (Temp(it).constraint_violation == 1)&&(Temp(jt).constraint_violation
== 1)&&(Temp(it).overalviolation < Temp(jt).overalviolation)

                        T_flag1 = 'non-dominated';

                    else
                        T_flag1 = 'dominated';

                        break;

                    end
                else
                    T_flag1 = 'dominated';

                    break;
                end
            end
        end

        C_7 = strcmp(T_flag1,'non-dominated');
        if C_7 == 1
            T_t=T_t + 1;
            if T_t<2
                Non_dominatedTemp = Temp(it);
            else
                Non_dominatedTemp(T_t) = Temp(it);

            end
        end
    end

    end

end
L_Non_dominatedTemp = length(Non_dominatedTemp);
if (Archivesize_max==1)&&(Nleaders_max==1)
    if Iter>10
        selection_method = 'Random';
    end
end
if L_Non_dominatedTemp > Archivesize_max
    C_10 = strcmp(selection_method , 'Random');
    if C_10 == 1
        index_select = randperm(Archivesize_max);
        Archive = Non_dominatedTemp(index_select(1));
        for i = 2:Archivesize_max
            Archive(i) = Non_dominatedTemp(index_select(i));
        end
    else
        Non_dominatedTemp =
aggeravated_fit_func(Non_dominatedTemp,selection_method,NumOFs,SwarmSize);
```

```
            Sorted_Non_dominatedTemp =
quicksort_fitness(Non_dominatedTemp,1,L_Non_dominatedTemp);

            L_Sorted_Non_dominatedTemp =length(Sorted_Non_dominatedTemp);
            Archive = Sorted_Non_dominatedTemp((L_Sorted_Non_dominatedTemp -
Archivesize_max +1) : L_Sorted_Non_dominatedTemp);

         end
      else
         if (newArchivemember_fromswarm >0)
            Non_dominatedTemp =
aggeravated_fit_func(Non_dominatedTemp,selection_method,NumOFs,SwarmSize);
            Sorted_Non_dominatedTemp =
quicksort_fitness(Non_dominatedTemp,1,L_Non_dominatedTemp);
            Archive = Sorted_Non_dominatedTemp;
         end


         Archive = Non_dominatedTemp;
      end
      L_Archive=length(Archive);
      for i=1:(L_Archive)
         for j= 1:(length(Pre_Archive))
            if  all(Archive(i).Position.descrete == Pre_Archive(j).Position.descrete) &&
all(Archive(i).Position.continuous == Pre_Archive(j).Position.continuous)

               Archive(i).archive_success = Archive(i).archive_success + 1;

               break

            end
         end
      end
      if (Archivesize_max > 1)&&(Nleaders_max==1)
         index_select_L = randperm(Nleaders_max);
         Leaders = Archive(index_select_L(1));
      else
         Num_leaders = min(min( Nleaders_max,L_Archive), ceil(leaders_percentage *
SwarmSize));
         Leaders = Archive((L_Archive - Num_leaders +1) : L_Archive);
      end
      if (L_Non_dominatedTemp > Archivesize_max) && (C_10==0)
         Final_non_dominated_set = Sorted_Non_dominatedTemp;
      else
         Final_non_dominated_set = Non_dominatedTemp;
      end
      %============== stopping criteria
      %==============================defined on OFs space
      if Iter>70
         manam=0;
```

```
end
n_Q_SIF =0;
for i=1:L_Archive
   if (Archive(i).archive_success - lag_SIF)>= 1
      n_Q_SIF = n_Q_SIF + 1;
   end
end
CR_SIF = n_Q_SIF/L_Archive;


nwe_cost_holder=[]; scaled_cost_holder=[];
if stopping_meth == 'Var_Ave'
   old_MEAN = new_MEAN;
   old_VAR = new_VAR;

   for i=1:L_Archive
      nwe_cost_holder(i,:) = Archive(i).Cost;
   end

   for j=1:(NumOFs-1)
      Min_j = min(nwe_cost_holder(:,j));
      Max_j = max(nwe_cost_holder(:,j));

      if (Max_j - Min_j)== 0

         scaled_cost_holder(:,j) = ones(L_Archive,1);
      else
         scaled_cost_holder(:,j) = (nwe_cost_holder(:,j) - Min_j)./(Max_j - Min_j);
      end
   end
   for j=1:(NumOFs-1)
      new_MEAN(j)=mean(scaled_cost_holder(:,j));
      new_VAR(j)=var(scaled_cost_holder(:,j));
   end
   if Iter > 2
      if old_VAR==0
         var_change_mat = new_VAR;
      else
         var_change_mat = abs((old_VAR - new_VAR)./old_VAR)*100;
      end
      Mean_var_change = mean(var_change_mat);
      Centroid_move = (sum((new_MEAN -old_MEAN).^2))^0.5;
      if (Centroid_move < 0.001)&&(Mean_var_change < 1)
         stopping_criteria_count = stopping_criteria_count+1;
      else
         stopping_criteria_count = 0;
      end
   end
   if (stopping_criteria_count < stopping_criteria_count_max)
      check_stop =1;
   else
```

```
        if Iter > (ceil(MaxIter*Iter_constdominancy)+1)
            check_stop = 0;
        end
    end
end
if stopping_meth == 'Con_Rat'
    n_Q =0;
    for i=1:L_Archive
        if (Archive(i).archive_success - lag)>= 1
            n_Q = n_Q + 1;
        end
    end
    CR = n_Q/L_Archive;
    if (CR < 0.6)
        check_stop =1;
    else
        if Iter > (ceil(MaxIter*Iter_constdominancy)+1)
            check_stop = 0;
        end
    end
end
if stopping_meth == 'Imp_Rat'
    n_Q =0;
    for i=1:L_Archive
        if ((Archive(i).archive_success - lag)>= 1)&& (Iter >16)
            n_Q = n_Q + 1;
        end
    end
    IR = c - n_Q;
    if (IR  > 0.66)
        check_stop =1;
    else
        if Iter > (ceil(MaxIter*Iter_constdominancy)+1)
            check_stop = 0;
        end
    end
end
if stopping_meth =='Com_Cri'
    n_Q =0; %***
    for i=1:L_Archive
        if (Archive(i).archive_success - lag)>= 1
            n_Q = n_Q + 1;
        end
    end
    CR = n_Q/L_Archive;

    old_MEAN = new_MEAN;
    old_VAR = new_VAR;
    %
    for i=1:L_Archive
        nwe_cost_holder(i,:) = Archive(i).Cost;
```

```
        end
        %
        for j=1:(NumOFs-1)
            Min_j = min(nwe_cost_holder(:,j));
            Max_j = max(nwe_cost_holder(:,j));

            if (Max_j - Min_j)==0
                scaled_cost_holder(:,j) = ones(L_Archive,1);
            else
                scaled_cost_holder(:,j) = (nwe_cost_holder(:,j) - Min_j)./(Max_j - Min_j);
            end
        end
        for j=1:(NumOFs-1)
            new_MEAN(j)=mean(scaled_cost_holder(:,j));
            new_VAR(j)=var(scaled_cost_holder(:,j));
        end
        if Iter > 2
            if old_VAR==0
                var_change_mat = new_VAR;
            else
                var_change_mat = abs((old_VAR - new_VAR)./old_VAR)*100;
            end
            Mean_var_change = mean(var_change_mat);
            Centroid_move = (sum((new_MEAN -old_MEAN).^2))^0.5;
            if (Centroid_move < 0.1)&&(Mean_var_change < 1)
                stopping_criteria_count = stopping_criteria_count+1;
            else
                stopping_criteria_count = 0;
            end
        end
        if (CR < 0.85)|| (stopping_criteria_count < stopping_criteria_count_max)
            check_stop =1;
        else
            if Iter > (ceil(MaxIter*Iter_constdominancy)+1)
                check_stop = 0;
            end
        end
    end
    Pre_Archive = [];
    Pre_Archive = Archive;
    Temp=[];Non_dominatedTemp=[]; Sorted_Non_dominatedTemp = [];


  end


end
```

Appendix B: ROI function MATLAB code

```
% this function calculates the Return on the Investment %ROI for DME
% process based on Douglas (1988) method and was written by Mazdak Shokrian

 function  roi_return = ROI(spreadvars,Y)
%+++++++++here all flows including heat flows are per
%hour
To_cond1=spreadvars(1);reflux1=spreadvars(2);Cp1=spreadvars(3);dH1=spreadvars(4);TtoC1=s
preadvars(5);
Tr1=spreadvars(6);To_cond2=spreadvars(7);reflux2=spreadvars(8);Cp2=spreadvars(9);dH2=spre
advars(10);
TtoC2=spreadvars(11);Tr2=spreadvars(12);Qex0=spreadvars(13);Ts2=spreadvars(14);Qex1=spre
advars(15);
Qex2=spreadvars(16);Ts4=spreadvars(17);Qex3=spreadvars(18);Qex4=spreadvars(19);Ftoboil1=
spreadvars(20);
Cpb1=spreadvars(21);Tboild1=spreadvars(22);TtoB1=spreadvars(23);DHtoexproc=spreadvars(2
4);Ftoboil2=spreadvars(25);
Cpb2=spreadvars(26);Tboild2=spreadvars(27);TtoB2=spreadvars(28);DHbuton=spreadvars(29);
Fboild2=spreadvars(30);
MProd=spreadvars(31);Ms17=spreadvars(32);Ms23=spreadvars(33);Ms24=spreadvars(34);Ms26
=spreadvars(35);
Ms13=spreadvars(36);Ms28=spreadvars(37);Ms30=spreadvars(38);Ms21=spreadvars(39);Ms19=
spreadvars(40);
Pe2=spreadvars(41);MWaste1=spreadvars(42);MWaste2=spreadvars(43);
%================================design DME
XTee1=spreadvars(44);XTee2=spreadvars(45);Fboild1=spreadvars(46);col1feedpress=spreadvars
(47); Reactorvolume = spreadvars(52); MFreshfeed = spreadvars(53);
Y1=Y(1);Y2=Y(2);

%================================design DME
MS = 1537.4;
Reactor = (MS/814)*(47*(Reactorvolume*264)^0.61); T_100_vessel =
(MS/280)*101.9*(2.5)^1.066*(45.93)^0.802*(2.18+3.67*1.15);
T_100_internals = (MS/280)* 4.7*(2.5)^1.55*(37.4)*(1.7+0.4+1);
T_101_vessel = (MS/280)*101.9*(2.5)^1.066*(52.45)^0.802*(2.18+3.67*1.15);
T_101_internals = (MS/280)* 4.7*(2.5)^1.55*(45.27)*(1.7+0+1);
```

```
%%%=========Condensers
    %+++firs Tower
    Q11=To_cond1*Cp1*(TtoC1-Tr1);
    Q21= reflux1*dH1;
    Qt1=Q11+Q21;
    t11=50-Q11/Qt1*(50-30);
    t21=30;%+Q21/Qt1*(50-30);
    A11=0.27777*Q11*abs(log((TtoC1-50)/(Tr1-t11))/(TtoC1-50-Tr1+t11))/(10*5.6782);
    A21=0.27777*Q21*abs(log((Tr1-t21)/(Tr1-t11))/(t11-t21))/(250*5.6782);
    A1=(A11)+(A21);                    %==================================design DME
condenserT1= (MS/280)*101.3*A1^0.65*(2.29+(0+1)*2.81);%floating head SS/CS
    %+++second Tower
    Q12=To_cond2*Cp2*(TtoC2-Tr2);
    Q22= reflux2*dH2;
    Qt2=Q12+Q22;
    t12=50-Q12/Qt2*(50-30);
    t22=30     A12=0.27777*Q12*abs(log((TtoC2-50)/(Tr2-t12))/(TtoC2-50-
Tr2+t12))/(10*5.6782);
    A22=0.27777*Q22*abs(log((Tr2-t22)/(Tr2-t12))/(t12-t22))/(250*5.6782);
    A2=(A12)+(A22);                    %==================================design DME
condenserT2= (MS/280)*101.3*A2^0.65*(2.29+(0+1)*2.81);%floating head SS/CS
%%%=========Boilers
     %+++firs Tower
     Qb11=Ftoboil1*Cpb1*(Tboild1-TtoB1);
     Qb12=DHtoexproc*Fboild1;
     Ab1=Qb11*abs(log((TtoB1-185.5)/(Tboild1-185.5))/(TtoB1-Tboild1))/(50*5.6782)+
Qb12/(185.5-Tboild1)/(250*5.6782);
BoilerT1= (MS/280)*101.3*Ab1^0.65*(2.29+(0+1.35)*2.81);%Kettle SS/CS
     %+++second Tower
     Qb21=Ftoboil2*Cpb2*(Tboild2-TtoB2);
     Qb22=DHbuton*Fboild2;
     Ab2=Qb21*abs(log((TtoB2-185.5)/(Tboild2-185.5))/(TtoB2-Tboild2))/(50*5.6782)+
Qb22/(185.5-Tboild2)/(250*5.6782);
BoilerT2= (MS/280)*101.3*Ab2^0.65*(2.29+(0+1.35)*2.81);%Kettle SS/CS
%%%==========Heat Exchangers

  %%%%=======hex 100
  %Ts2  temp of stream 2
DTmex0 = abs ((Ts2-152.5)/log((Ts2-253)/(152.5-253)));
Aex0 =Qex0/(DTmex0*113.564);
EX0= (MS/280)*101.3*Aex0^0.65*(2.29+(0.1+1)*2.81);%Floating head SS/CS
%%%%=======hex 101
DTmex1 = abs((152.5-220)/log((152.5-253)/(220-253)));
Aex1 =Qex1/(DTmex1*113.564*0.5);
EX1= (MS/280)*101.3*Aex1^0.65*(2.29+(0.1+1)*2.81);%Floating head SS/CS
%%%%=======hex 102
DTmex2 = abs((Ts4-50 -177.1+30)/log((Ts4-50)/(177.1-30)));
Aex2 =Qex2/(DTmex2*113.564);
EX2= (MS/280)*101.3*Aex2^0.65*(2.29+(0.1+1)*2.81);%Floating head SS/CS
%%%%=======hex 103
DTmex3 = abs((177.1-50 -110+30)/log((177.1-50)/(110-30)));
```

```
Aex3 =Qex3/(DTmex3*113.564*50/20);
EX3= (MS/280)*101.3*Aex3^0.65*(2.29+(0.1+1)*2.81);%Floating head SS/CS
%%%%=======hex 104
DTmex4 = abs((165-50 -50+30)/log((165-50)/(50-30)));
Aex4 =Qex4/(DTmex4*113.564*50/20);
EX4= (MS/280)*101.3*Aex4^0.65*(2.29+(0+1)*2.81);%Floating head SS/CS



captalcost =
EX0+EX1+EX2+EX3+Y1*EX4+BoilerT1+Y1*BoilerT2+condenserT1+Y1*condenserT2+Y1*
T_101_internals+Y1*T_101_vessel+T_100_internals+T_100_vessel+Reactor;
%================================design DME
AnnRaw =0.9946*MFreshfeed*8150*0.294;  % Kmole of the Raw material per hr*8150*price
per Kmole???;

AnnUtil =
(Ms17+Ms23+Ms24+Y1*(Ms26+Ms13))*8150*0.148/1000+(Ms28+Ms30)*8150*0.02997+(Ms
21+Y1*Ms19)*8150*0.02954+ (2.486+Pe2)*8150*0.0717;

AnnRevenue = MProd*8150*1.17;   %Kmole of the product per hr*8150*price per Kmole
Annwastetreat = (Y2*MWaste1+Y1*MWaste2)*8150*0.036;
Anncaptalcost=captalcost/3;
Deprec = 0.181*captalcost;
Ann_Prof = 0.52*((0.974*AnnRevenue-1.03*(AnnRaw +AnnUtil)-0.186*captalcost -
2.13*100000 - Annwastetreat)-Deprec);
Tot_Inv = 2.36*captalcost;
roi_return = 100*Ann_Prof/Tot_Inv;
```

Appendix B: MATLAB code of the function for getting sustainability information from users

```
% this functions takes the required information about the process through GUIs and creates a
matrix for sustainability evaluation
clear
clc
load susfactors;

%%%++++++++++++++++++++++++++++++++ Environemental
Burden================================================================
%Atmospheric_Acidification
Number_total_diffrent_chemi =0;
chemi_counter = 0;
AA_list = Susfactors.Atmospheric_Acidification.name;
AA_potency = Susfactors.Atmospheric_Acidification.potencyfactors;
AA_list(7) =[];AA_potency(7) =[];
[s,v] = listdlg('PromptString','Atmospheric_Acidification
Chem','SelectionMode','single','ListString',AA_list);
N = length(s);
AA=[];
if N >= 1
   AA.name = AA_list(s);
   AA.potency = AA_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   AA.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      AA.ID = Id;
   end
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
   chemi_counter = chemi_counter+1;
end
```

```
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Atmospheric_Acidification
Chem','SelectionMode','single','ListString',AA_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            AA(chemi_counter).name = AA_list(s);
            AA(chemi_counter).potency = AA_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            AA(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               AA(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end

      case 'No'
         f=0;

         break
      case 'Cancel'

         f=0;

         break
      otherwise
                  f=0;

   end


end

%+++=++++++++++++++++++++++++++++++++++++++++
%Global_Warming
chemi_counter = 0;
GW_list = Susfactors.AGlobal_warming.name;
GW_potency = Susfactors.AGlobal_warming.potency;
GW_list(676) =[];GW_potency(676) =[];
[s,v] = listdlg('PromptString','Global Warming
Chem','SelectionMode','single','ListString',GW_list);
```

```
N = length(s);
GW=[];
if N >= 1
   GW.name = GW_list(s);
   GW.potency=GW_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   GW.ID = Id;
   chemi_counter = chemi_counter+1;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      GW.ID = Id;
   end
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Global Warming
Chem','SelectionMode','single','ListString',GW_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            GW(chemi_counter).name = GW_list(s);
            GW(chemi_counter).potency=GW_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            GW(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               GW(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         %disp('Method is nearest')
         f=0;
         break
      otherwise
```

```
        %disp('Unknown method.')
        f=0;
    end
end
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
%Stratospheric Ozone Depletion  SOD
chemi_counter = 0;
SOD_list = Susfactors.StratosphericOzoneDepletion.name;
SOD_potency = Susfactors.StratosphericOzoneDepletion.potency;
SOD_list(59) =[];SOD_potency(59) =[];
[s,v] = listdlg('PromptString','Stratospheric Ozone Depletion
Chem','SelectionMode','single','ListString',SOD_list);
N = length(s);
SOD=[];
if N >= 1
    SOD.name = SOD_list(s);
    SOD.potency =SOD_potency(s);
    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
    SOD.ID = Id;
    chemi_counter = chemi_counter+1;
    while isempty(Id);
        Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
        SOD.ID = Id;
    end
    if Id > Number_total_diffrent_chemi
        Number_total_diffrent_chemi =Id;
    end
end
f=1;
while f==1
    button = questdlg('is any more component in this category?');
    switch button
        case 'Yes'
            f=1;
            [s,v] = listdlg('PromptString','Stratospheric Ozone Depletion
Chem','SelectionMode','single','ListString',SOD_list);
            N = length(s);
            if N >= 1
                chemi_counter = chemi_counter+1;
                SOD(chemi_counter).name = SOD_list(s);
                SOD(chemi_counter).potency =SOD_potency(s);
                Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
                SOD(chemi_counter).ID = Id;
                while isempty(Id);
                    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                    SOD(chemi_counter).ID = Id;
                end
```

```
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         %disp('Method is nearest')
         f=0;
         break
      otherwise
         %disp('Unknown method.')
         f=0;
   end
end
%+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
%Photochemical smog Formation  PSF
chemi_counter = 0;
PSF_list = Susfactors.PhotochemicalsmogFormation.name;
PSF_potency = Susfactors.PhotochemicalsmogFormation.potency;
PSF_list(105) =[];PSF_potency(105) =[];
[s,v] = listdlg('PromptString','Photochemical smog Formation
Chem','SelectionMode','single','ListString',PSF_list);
N = length(s);
PSF=[];
if N >= 1
   PSF.name = PSF_list(s);
   PSF.potency =PSF_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   PSF.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      PSF.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Photochemical smog Formation
Chem','SelectionMode','single','ListString',PSF_list);
         N = length(s);
```

```
        if N >= 1
            chemi_counter = chemi_counter+1;
            PSF(chemi_counter).name = PSF_list(s);
            PSF(chemi_counter).potency =PSF_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            PSF(chemi_counter).ID = Id;
            while isempty(Id);
                Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                PSF(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
                Number_total_diffrent_chemi =Id;
            end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'
                f=0;
        break
    otherwise
                f=0;
  end
end
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
%Aquatic Acidification  AqAc
chemi_counter = 0;
AqAc_list = Susfactors.AquaticAcidification.name;
AqAc_potency = Susfactors.AquaticAcidification.potency;
AqAc_list(5) =[];AqAc_potency(5) =[];
[s,v] = listdlg('PromptString','Aquatic Acidification
Chem','SelectionMode','single','ListString',AqAc_list);
N = length(s);
AqAc=[];
if N >= 1
   AqAc.name = AqAc_list(s);
   AqAc.potency =AqAc_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   AqAc.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      AqAc.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
```

```
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Aquatic Acidification
Chem','SelectionMode','single','ListString',AqAc_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            AqAc(chemi_counter).name = AqAc_list(s);
            AqAc(chemi_counter).potency =AqAc_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            AqAc(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               AqAc(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
          f=0;
         break
      otherwise
                  f=0;
   end
end
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++
%Aquatic Oxygen Demand  AOD
chemi_counter = 0;
AOD_list = Susfactors.AquaticOxygenDemand.name;
AOD_potency = Susfactors.AquaticOxygenDemand.potency;
AOD_list(15) =[];AOD_potency(15) =[];
[s,v] = listdlg('PromptString','Aquatic Oxygen Demand
Chem','SelectionMode','single','ListString',AOD_list);
N = length(s);
AOD=[];
if N >= 1
   AOD.name = AOD_list(s);
   AOD.potency =AOD_potency(s);
```

```
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   AOD.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      AOD.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Aquatic Oxygen Demand
Chem','SelectionMode','single','ListString',AOD_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            AOD(chemi_counter).name = AOD_list(s);
            AOD(chemi_counter).potency =AOD_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            AOD(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               AOD(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         %disp('Method is nearest')
         f=0;
         break
      otherwise
         %disp('Unknown method.')
         f=0;
   end
end
```

```
%++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+

%Ecotoxicity Aquatic Life   EA
chemi_counter = 0;
EA_list = Susfactors.Ecotoxicity_AquaticLife.name;
EA_potency = Susfactors.Ecotoxicity_AquaticLife.potency;
EA_list(30) =[];EA_potency(30) =[];
[s,v] = listdlg('PromptString','Ecotoxicity Aquatic Life
Chem','SelectionMode','single','ListString',EA_list);
N = length(s);
EA=[];
if N >= 1
   EA.name = EA_list(s);
   EA.potency =EA_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   EA.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      EA.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Ecotoxicity Aquatic Life
Chem','SelectionMode','single','ListString',EA_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            EA(chemi_counter).name = EA_list(s);
            EA(chemi_counter).potency =EA_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            EA(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               EA(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
```

```
            end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'

        f=0;
        break
    otherwise
        f=0;
  end
end
%+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

%Eutrophication   Eutro
chemi_counter = 0;
Eutro_list = Susfactors.Eutrophication.name;
Eutro_potency = Susfactors.Eutrophication.potency;
Eutro_list(9) =[];Eutro_potency(9) =[];
[s,v] = listdlg('PromptString','Eutrophication
Chem','SelectionMode','single','ListString',Eutro_list);
N = length(s);
Eutro=[];
if N >= 1
   Eutro.name = Eutro_list(s);
   Eutro.potency =Eutro_potency(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Eutro.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Eutro.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Eutrophication
Chem','SelectionMode','single','ListString',Eutro_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
```

```
            Eutro(chemi_counter).name = Eutro_list(s);
            Eutro(chemi_counter).potency =Eutro_potency(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Eutro(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Eutro(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'

         f=0;
         break
      otherwise
         f=0;
   end
end
%%%++++++++++++++++++++++++++++++++ Health impacts
================================================================
%Carcinogen  Carc
chemi_counter = 0;
Carc_list = Susfactors.Carcinogen.name;
Carc_hazardlevel = Susfactors.Carcinogen.hazardlevel;
Carc_cancer_risk = Susfactors.Carcinogen.cancer_risk;
Carc_list(9) =[];Carc_cancer_risk(9) =[];Carc_hazardlevel(9) =[];
[s,v] = listdlg('PromptString','Carcinogen','SelectionMode','single','ListString',Carc_list);
N = length(s);
Carc=[];
if N >= 1
   Carc.name = Carc_list(s);
   Carc.risk =Carc_cancer_risk(s);
   Carc.hazardlevel =Carc_hazardlevel(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Carc.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Carc.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
```

```
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Carcinogen','SelectionMode','single','ListString',Carc_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            Carc(chemi_counter).name = Carc_list(s);
            Carc(chemi_counter).risk =Carc_cancer_risk(s);
            Carc(chemi_counter).hazardlevel =Carc_hazardlevel(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Carc(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Carc(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
                  f=0;
         break
      otherwise
         f=0;
   end
end
%%%suspected_immunotoxicants
chemi_counter = 0;
susimm_list = Susfactors.suspected_immunotoxicants.name;
susimm_score = Susfactors.suspected_immunotoxicants.score;
susimm_list(425) =[];susimm_score(425) =[];
[s,v] =
listdlg('PromptString','suspected_immunotoxicants','SelectionMode','single','ListString',susimm_li
st);
N = length(s);
susimm=[];
if N >= 1
   susimm.name = susimm_list(s);
   susimm.score =susimm_score(s);
```

```
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   susimm.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      susimm.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','suspected_immunotoxicants','SelectionMode','single','ListString',susimm_li
st);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            susimm(chemi_counter).name = susimm_list(s);
            susimm(chemi_counter).score =susimm_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            susimm(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               susimm.ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
                  f=0;
         break
      otherwise
                  f=0;
   end
end
%%%%% Neurotoxicity_hazard
```

```
chemi_counter = 0;
Neuro_list = Susfactors.Neurotoxicity_hazard.name;
Neuro_score = Susfactors.Neurotoxicity_hazard.score;
Neuro_list(1178) =[];Neuro_score(1178) =[];
[s,v] =
listdlg('PromptString','Neurotoxicity_hazard','SelectionMode','single','ListString',Neuro_list);
N = length(s);
Neuro=[];
if N >= 1
   Neuro.name = Neuro_list(s);
   Neuro.score =Neuro_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Neuro.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Neuro.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','Neurotoxicity_hazard','SelectionMode','single','ListString',Neuro_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            Neuro(chemi_counter).name = Neuro_list(s);
            Neuro(chemi_counter).score =Neuro_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Neuro(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Neuro(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
```

```
        break
    case 'Cancel'
            f=0;
        break
    otherwise
            f=0;
  end
end
%%%developemental_toxicant   devel
chemi_counter = 0;
devel_list = Susfactors.develepemental_toxicant.name;
devel_score = Susfactors.develepemental_toxicant.score;
devel_list(799) =[];devel_score(799) =[];
[s,v] =
listdlg('PromptString','developemental_toxicant','SelectionMode','single','ListString',devel_list);
N = length(s);
devel=[];
if N >= 1
  devel.name = devel_list(s);
  devel.score =devel_score(s);
  Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
  devel.ID = Id;
  chemi_counter = chemi_counter+1;
  while isempty(Id);
    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
    devel.ID = Id;
  end
  if Id > Number_total_diffrent_chemi
    Number_total_diffrent_chemi =Id;
  end
end
f=1;
while f==1
  button = questdlg('is any more component in this category?');
  switch button
    case 'Yes'
      f=1;
      [s,v] =
listdlg('PromptString','developemental_toxicant','SelectionMode','single','ListString',devel_list);
      N = length(s);
      if N >= 1
        chemi_counter = chemi_counter+1;
        devel(chemi_counter).name = devel_list(s);
        devel(chemi_counter).score =devel_score(s);
        Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
        devel(chemi_counter).ID = Id;
        while isempty(Id);
```

```
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
            devel(chemi_counter).ID = Id;
          end
          if Id > Number_total_diffrent_chemi
            Number_total_diffrent_chemi =Id;
          end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'
                f=0;
        break
    otherwise

        f=0;
  end
end
%%%%%%%Reproductive_toxicant   Repro
chemi_counter = 0;
Repro_list = Susfactors.Reproductive_toxicant.name;
Repro_score = Susfactors.Reproductive_toxicant.score;
Repro_list(355) =[];Repro_score(355) =[];
[s,v] =
listdlg('PromptString','Reproductive_toxicant','SelectionMode','single','ListString',Repro_list);
N = length(s);
Repro=[];
if N >= 1
  Repro.name = Repro_list(s);
  Repro.score =Repro_score(s);
  Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
  Repro.ID = Id;
  while isempty(Id);
    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
    Repro.ID = Id;
  end
  chemi_counter = chemi_counter+1;
  if Id > Number_total_diffrent_chemi
    Number_total_diffrent_chemi =Id;
  end
end
f=1;
while f==1
  button = questdlg('is any more component in this category?');
  switch button
    case 'Yes'
      f=1;
```

```
        [s,v] =
listdlg('PromptString','Reproductive_toxicant','SelectionMode','single','ListString',Repro_list);
        N = length(s);
        if N >= 1
            chemi_counter = chemi_counter+1;
            Repro(chemi_counter).name = Repro_list(s);
            Repro(chemi_counter).score =Repro_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Repro(chemi_counter).ID = Id;
            while isempty(Id);
                Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                Repro(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
                Number_total_diffrent_chemi =Id;
            end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'

        break
    otherwise
                f=0;
  end
end
%%%%kidney_toxicant
chemi_counter = 0;
kidney_list = Susfactors.kidney_toxicant.name;
kidney_score = Susfactors.kidney_toxicant.score;
kidney_list(425) =[];kidney_score(425) =[];
[s,v] = listdlg('PromptString','kidney_toxicant','SelectionMode','single','ListString',kidney_list);
N = length(s);
kidney=[];
if N >= 1
   kidney.name = kidney_list(s);
   kidney.score =kidney_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   kidney.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      kidney(chemi_counter).ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
```

```
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','kidney_toxicant','SelectionMode','single','ListString',kidney_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            kidney(chemi_counter).name = kidney_list(s);
            kidney(chemi_counter).score =kidney_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            kidney(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               kidney(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
                  f=0;
         break
      otherwise
                  f=0;
   end
end

%%%%Respiratory   chemi_counter = 0;
Resp_list = Susfactors.Respiratory.name;
Resp_score = Susfactors.Respiratory.score;
Resp_list(851) =[];Resp_score(851) =[];
[s,v] = listdlg('PromptString','Respiratory','SelectionMode','single','ListString',Resp_list);
N = length(s);
Resp=[];
if N >= 1
   Resp.name = Resp_list(s);
   Resp.score =Resp_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Resp.ID = Id;
```

```
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Resp.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Respiratory','SelectionMode','single','ListString',Resp_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            Resp(chemi_counter).name = Resp_list(s);
            Resp(chemi_counter).score =Resp_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Resp(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Resp(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
                  f=0;
         break
      otherwise
         f=0;
   end
end
%%%%%Cardiovascular_toxicant
chemi_counter = 0;
Cardio_list = Susfactors.Cardiovascular_toxicant.name;
Cardio_score = Susfactors.Cardiovascular_toxicant.score;
Cardio_list(676) =[];Cardio_score(676) =[];
[s,v] =
listdlg('PromptString','Cardiovascular_toxicant','SelectionMode','single','ListString',Cardio_list);
```

```
N = length(s);
Cardio=[];
if N >= 1
   Cardio.name = Cardio_list(s);
   Cardio.score =Cardio_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Cardio.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Cardio.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','Cardiovascular_toxicant','SelectionMode','single','ListString',Cardio_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            Cardio(chemi_counter).name = Cardio_list(s);
            Cardio(chemi_counter).score =Cardio_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Cardio(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Cardio(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         f=0;
         break
      otherwise
         f=0;
```

```
   end
end

%%%%%Suspected_endocrine_toxicant
chemi_counter = 0;
endocrine_list = Susfactors.Suspected_endocrine_toxicant.name;
endocrine_score = Susfactors.Suspected_endocrine_toxicant.score;
endocrine_list(311) =[];endocrine_score(311) =[];
[s,v] =
listdlg('PromptString','Suspected_endocrine_toxicant','SelectionMode','single','ListString',endocri
ne_list);
N = length(s);
endocrine=[];
if N >= 1
   endocrine.name = endocrine_list(s);
   endocrine.score =endocrine_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   endocrine.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      endocrine.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','Suspected_endocrine_toxicant','SelectionMode','single','ListString',endocri
ne_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            endocrine(chemi_counter).name = endocrine_list(s);
            endocrine(chemi_counter).score =endocrine_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            endocrine(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               endocrine(chemi_counter).ID = Id;
            end
```

```
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         f=0;
         break
      otherwise
         f=0;
   end
end

%%%%%%Gastrointestinal_liver_toxicant

chemi_counter = 0;
Gastro_list = Susfactors.Gastrointestinal_liver_toxicant.name;
Gastro_score = Susfactors.Gastrointestinal_liver_toxicant.score;
Gastro_list(905) =[];Gastro_score(905) =[];
[s,v] =
listdlg('PromptString','Gastrointestinal_liver_toxicant','SelectionMode','single','ListString',Gastro_
list);
N = length(s);
Gastro=[];
if N >= 1
   Gastro.name = Gastro_list(s);
   Gastro.score =Gastro_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
   Gastro.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Gastro.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','Gastrointestinal_liver_toxicant','SelectionMode','single','ListString',Gastro_
list);
```

```
        N = length(s);
        if N >= 1
            chemi_counter = chemi_counter+1;
            Gastro(chemi_counter).name = Gastro_list(s);
            Gastro(chemi_counter).score =Gastro_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Gastro(chemi_counter).ID = Id;
            while isempty(Id);
                Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                Gastro(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
                Number_total_diffrent_chemi =Id;
            end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'
        f=0;
        break
    otherwise
        f=0;
  end
end
%%%%%Musculoskeletal_toxicant

chemi_counter = 0;
skelet_list = Susfactors.Musculoskeletal_toxicant.name;
skelet_score = Susfactors.Musculoskeletal_toxicant.score;
skelet_list(35) =[];skelet_score(35) =[];
[s,v] =
listdlg('PromptString','Musculoskeletal_toxicant','SelectionMode','single','ListString',skelet_list);
N = length(s);
skelet=[];
if N >= 1
  skelet.name = skelet_list(s);
  skelet.score =skelet_score(s);
  Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
  skelet.ID = Id;
  while isempty(Id);
    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
    skelet.ID = Id;
  end
  chemi_counter = chemi_counter+1;
  if Id > Number_total_diffrent_chemi
    Number_total_diffrent_chemi =Id;
```

```
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] =
listdlg('PromptString','Musculoskeletal_toxicant','SelectionMode','single','ListString',skelet_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            skelet(chemi_counter).name = skelet_list(s);
            skelet(chemi_counter).score =skelet_score(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            skelet(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               skelet(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         f=0;
         break
      otherwise
         f=0;
   end
end
%%%%%Skin_sense_organ_toxicant
chemi_counter = 0;
Skin_list = Susfactors.Skin_sense_organ_toxicant.name;
Skin_score = Susfactors.Skin_sense_organ_toxicant.score;
Skin_list(865) =[];Skin_score(865) =[];
[s,v] =
listdlg('PromptString','Skin_sense_organ_toxicant','SelectionMode','single','ListString',Skin_list);
N = length(s);
Skin=[];
if N >= 1
   Skin.name = Skin_list(s);
   Skin.score =Skin_score(s);
   Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it for
the same item in other categories \n');
```

```
    Skin.ID = Id;
    while isempty(Id);
        Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
        Skin.ID = Id;
    end
    chemi_counter = chemi_counter+1;
    if Id > Number_total_diffrent_chemi
        Number_total_diffrent_chemi =Id;
    end
end
f=1;
while f==1
    button = questdlg('is any more component in this category?');
    switch button
        case 'Yes'
            f=1;
            [s,v] =
listdlg('PromptString','Skin_sense_organ_toxicant','SelectionMode','single','ListString',Skin_list);
            N = length(s);
            if N >= 1
                chemi_counter = chemi_counter+1;
                Skin(chemi_counter).name = Skin_list(s);
                Skin(chemi_counter).score =Skin_score(s);
                Id = input('Enter an ID for the selected item as the rank it was first enterted \n  and
insert it for the same item in other categories \n');
                Skin(chemi_counter).ID = Id;
                while isempty(Id);
                    Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                    Skin(chemi_counter).ID = Id;
                end
                if Id > Number_total_diffrent_chemi
                    Number_total_diffrent_chemi =Id;
                end
            end
        case 'No'
            f=0;
            break
        case 'Cancel'
            f=0;
            break
        otherwise
            f=0;
    end
end

%==================================Safety=============================
==================

%%%Flash point  Flash
```

```
chemi_counter = 0;
Flash_list = Susfactors.Flashpoint.name;
Flash_point = Susfactors.Flashpoint.degreecelcyus;
 [s,v] = listdlg('PromptString','select any existing
component','SelectionMode','single','ListString',Flash_list);
N = length(s);
Flash=[];
if N >= 1
   Flash.name = Flash_list(s);
   Flash.point =Flash_point(s);
   Id = input('Enter an ID for the selected item as the rank it was first enterted \n  and insert it for
the same item in other categories \n');
   Flash.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      Flash.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
         f=1;
         [s,v] = listdlg('PromptString','Eutrophication
Chem','SelectionMode','single','ListString',Flash_list);
         N = length(s);
         if N >= 1
            chemi_counter = chemi_counter+1;
            Flash(chemi_counter).name = Flash_list(s);
            Flash(chemi_counter).point =Flash_piont(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            Flash(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               Flash(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
```

```
        case 'Cancel'
            f=0;
            break
        otherwise
            f=0;
    end
end
%%%%%%LEL_UEL    LEL_UEL
chemi_counter = 0;
LEL_UEL_list = Susfactors.LEL_UEL.name;
LEL = Susfactors.LEL_UEL.LEL;
UEL = Susfactors.LEL_UEL.UEL;
LEL_UEL_difference= Susfactors.LEL_UEL.Unknown;
[s,v] = listdlg('PromptString','select any existing
component','SelectionMode','single','ListString',LEL_UEL_list);
N = length(s);
LEL_UEL=[];
if N >= 1
    LEL_UEL.name = LEL_UEL_list(s);
    LEL_UEL.LEL =LEL(s);
    LEL_UEL.UEL =UEL(s);
    LEL_UEL.difference =LEL_UEL_difference(s);
    Id = input('Enter an ID for the selected item as the rank it was first enterted \n  and insert it for
the same item in other categories \n');
    LEL_UEL.ID = Id;
    while isempty(Id);
        Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
        LEL_UEL.ID = Id;
    end
    chemi_counter = chemi_counter+1;
    if Id > Number_total_diffrent_chemi
        Number_total_diffrent_chemi =Id;
    end
end
f=1;
while f==1
    button = questdlg('is any more component in this category?');
    switch button
        case 'Yes'
            f=1;
            [s,v] = listdlg('PromptString','Eutrophication
Chem','SelectionMode','single','ListString',LEL_UEL_list);
            N = length(s);
            if N >= 1
                chemi_counter = chemi_counter+1;
                LEL_UEL(chemi_counter).name = LEL_UEL_list(s);
                LEL_UEL(chemi_counter).LEL =LEL(s);
                LEL_UEL(chemi_counter).UEL =UEL(s);
                LEL_UEL(chemi_counter).difference =LEL_UEL_difference(s);
```

```
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            LEL_UEL(chemi_counter).ID = Id;
            while isempty(Id);
               Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
               LEL_UEL(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
               Number_total_diffrent_chemi =Id;
            end
         end
      case 'No'
         f=0;
         break
      case 'Cancel'
         f=0;
         break
      otherwise
         f=0;
   end
end
%%%%%==========TLV
chemi_counter = 0;
TLV_list = Susfactors.TLV.name;
TLV_num = Susfactors.TLV.tlvppm;
[s,v] = listdlg('PromptString','select any existing
component','SelectionMode','single','ListString',TLV_list);
N = length(s);
TLV=[];
if N >= 1
   TLV.name = TLV_list(s);
   TLV.num =TLV_num(s);
   Id = input('Enter an ID for the selected item as the rank it was first enterted \n  and insert it for
the same item in other categories \n');
   TLV.ID = Id;
   while isempty(Id);
      Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert it
for the same item in other categories \n');
      TLV.ID = Id;
   end
   chemi_counter = chemi_counter+1;
   if Id > Number_total_diffrent_chemi
      Number_total_diffrent_chemi =Id;
   end
end
f=1;
while f==1
   button = questdlg('is any more component in this category?');
   switch button
      case 'Yes'
```

```
        f=1;
        [s,v] = listdlg('PromptString','Eutrophication
Chem','SelectionMode','single','ListString',TLV_list);
        N = length(s);
        if N >= 1
            chemi_counter = chemi_counter+1;
            TLV(chemi_counter).name = TLV_list(s);
            TLV(chemi_counter).num =TLV_num(s);
            Id = input('Enter an ID for the selected item as the rank it was first entered \n  and insert
it for the same item in other categories \n');
            TLV(chemi_counter).ID = Id;
            while isempty(Id);
                Id = input('Enter an ID for the selected item as the rank it was first entered \n  and
insert it for the same item in other categories \n');
                TLV(chemi_counter).ID = Id;
            end
            if Id > Number_total_diffrent_chemi
                Number_total_diffrent_chemi =Id;
            end
        end
    case 'No'
        f=0;
        break
    case 'Cancel'
        f=0;
        break
    otherwise
        f=0;
  end
end

%============creating the SUS_EVAL==========================
SUS_MAT=[];tt=0;OFsCount=zeros(1,12);
NumOFs= 6health_counter=0;

health_social_environmenal = questdlg('do you want to evaluate environmental burden or social
or health metrics?');

if strcmpi(health_social_environmenal,'Yes')
  for i=1:Number_total_diffrent_chemi

    flag=0;
    if isempty(AA)
        SUS_MAT(i,1)=0;
        flag=0;
    else
        for j=1:length(AA)
            if AA(j).ID==i
                tt=tt+1;
                all_selected_chemicals(tt,1) =AA(j).name;
                all_selected_chemicals_ID(tt,1) =AA(j).ID;
```

```
            SUS_MAT(i,1)=AA(j).potency;
            flag=1;
            OFsCount(1,1)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,1)=0;
    end
    flag=0;
end

%Global Warming ==2
if isempty(GW)
    SUS_MAT(i,2)=0;
else
    for j=1:length(GW)
        if GW(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =GW(j).name;
            all_selected_chemicals_ID(tt,1) =GW(j).ID;
            SUS_MAT(i,2)=GW(j).potency;
            flag=1;
            % NumOFs=NumOFs+1;
            OFsCount(1,2)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,2)=0;
    end
    flag=0;
end
%Stratospheric Ozone Depletion==3
if isempty(SOD)
    SUS_MAT(i,3)=0;
else
    for j=1:length(SOD)
        if SOD(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =SOD(j).name;
            all_selected_chemicals_ID(tt,1) =SOD(j).ID;
            SUS_MAT(i,3)=SOD(j).potency;
            flag=1;

            OFsCount(1,3)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,3)=0;
```

```
    end
    flag=0;
end

%Photochemical smog Formation==4
if isempty(PSF)
    SUS_MAT(i,4)=0;
else
    for j=1:length(PSF )
        if PSF(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =PSF(j).name;
            all_selected_chemicals_ID(tt,1) =PSF(j).ID;
            SUS_MAT(i,4)=PSF(j).potency;
            flag=1;

            OFsCount(1,4)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,4)=0;
    end
    flag=0;
end
%Aquatic Acidification  AqAc==5
if isempty(AqAc)
    SUS_MAT(i,5)=0;
else
    for j=1:length(AqAc )
        if AqAc(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =AqAc(j).name;
            all_selected_chemicals_ID(tt,1) =AqAc(j).ID;
            SUS_MAT(i,5)=AqAc(j).potency;
            flag=1;

            OFsCount(1,5)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,5)=0;
    end
    flag=0;
end
%Aquatic Oxygen Demand  AOD==6
if isempty(AOD)
    SUS_MAT(i,6)=0;
else
    for j=1:length(AOD)
```

```
        if AOD(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =AOD(j).name;
            all_selected_chemicals_ID(tt,1) =AOD(j).ID;
            SUS_MAT(i,6)=AOD(j).potency;
            flag=1;
            OFsCount(1,6)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,6)=0;
    end
    flag=0;
end
%Ecotoxicity Aquatic Life   EA==7
if isempty(EA)
    SUS_MAT(i,7)=0;
else
    for j=1:length(EA)
        if EA(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =EA(j).name;
            all_selected_chemicals_ID(tt,1) =EA(j).ID;
            SUS_MAT(i,7)=EA(j).potency;
            flag=1;
                        OFsCount(1,7)=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,7)=0;
    end
    flag=0;
end
%Eutrophication   Eutro==8
if isempty(Eutro)
    SUS_MAT(i,8)=0;
else
    for j=1:length(Eutro)
        if Eutro(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Eutro(j).name;
            all_selected_chemicals_ID(tt,1) =Eutro(j).ID;
            SUS_MAT(i,8)=Eutro(j).potency;
            flag=1;
                        OFsCount(1,8)=1;
            break
        end
    end
    if flag ==0
```

```
        SUS_MAT(i,8)=0;
    end
    flag=0;
end
%++++++++++++++++++++++++++++++++++++++++++++++++++++health
%Carcinogen  Carc
if isempty(Carc)
    SUS_MAT(i,9)=0;
else
    for j=1:length(Carc)
        if Carc(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Carc(j).name;
            all_selected_chemicals_ID(tt,1) =Carc(j).ID;
            SUS_MAT(i,9)=Carc(j).risk;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,9)=0;
    end
    flag=0;
end
%%%suspected_immunotoxicants   susimm
if isempty(susimm)
    SUS_MAT(i,10)=0;
else
    for j=1:length(susimm)
        if susimm(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =susimm(j).name;
            all_selected_chemicals_ID(tt,1) =susimm(j).ID;
            SUS_MAT(i,10)=susimm(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,10)=0;
    end
    flag=0;
end
%%%%%% Neurotoxicity_hazard Neuro
if isempty(Neuro)
    SUS_MAT(i,11)=0;
else
    for j=1:length(Neuro)
        if Neuro(j).ID==i
```

```
            tt=tt+1;
            all_selected_chemicals(tt,1) =Neuro(j).name;
            all_selected_chemicals_ID(tt,1) =Neuro(j).ID;
            SUS_MAT(i,11)=Neuro(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,11)=0;
    end
    flag=0;
end
%%%developemental_toxicant   devel
if isempty(devel)
    SUS_MAT(i,12)=0;
else
    for j=1:length(devel)
        if devel(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =devel(j).name;
            all_selected_chemicals_ID(tt,1) =devel(j).ID;
            SUS_MAT(i,12)=devel(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,12)=0;
    end
    flag=0;
end
%%%%%%%Reproductive_toxicant   Repro
if isempty(Repro)
    SUS_MAT(i,13)=0;
else
    for j=1:length(Repro)
        if Repro(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Repro(j).name;
            all_selected_chemicals_ID(tt,1) =Repro(j).ID;
            SUS_MAT(i,13)=Repro(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,13)=0;
```

```
    end
    flag=0;
end
%%%%kidney_toxicant  kidney
if isempty(kidney)
    SUS_MAT(i,14)=0;
else
    for j=1:length(kidney)
        if kidney(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =kidney(j).name;
            all_selected_chemicals_ID(tt,1) =kidney(j).ID;
            SUS_MAT(i,14)=kidney(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,14)=0;
    end
    flag=0;
end
%%%%Respiratory   Resp
if isempty(Resp)
    SUS_MAT(i,15)=0;
else
    for j=1:length(Resp)
        if Resp(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Resp(j).name;
            all_selected_chemicals_ID(tt,1) =Resp(j).ID;
            SUS_MAT(i,15)=Resp(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,15)=0;
    end
    flag=0;
end
%%%%%Cardiovascular_toxicant   Cardio
if isempty(Cardio)
    SUS_MAT(i,16)=0;
else
    for j=1:length(Cardio)
        if Cardio(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Cardio(j).name;
```

```
            all_selected_chemicals_ID(tt,1) =Cardio(j).ID;
            SUS_MAT(i,16)=Cardio(j).score;
            flag=1;
            health_counter=health_counter+1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,16)=0;
    end
    flag=0;
end
%%%%%Suspected_endocrine_toxicant   endocrine
if isempty(endocrine)
    SUS_MAT(i,17)=0;
else
    for j=1:length(endocrine)
        if endocrine(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =endocrine(j).name;
            all_selected_chemicals_ID(tt,1) =endocrine(j).ID;
            SUS_MAT(i,17)=endocrine(j).score;
            flag=1;
            break   % this would prevent repetetive selection comming to the SUS_MAT
            health_counter=health_counter+1;
        end
    end
    if flag ==0
        SUS_MAT(i,17)=0;
    end
    flag=0;
end

%%%%%Gastrointestinal_liver_toxicant   Gastro
if isempty(Gastro)
    SUS_MAT(i,18)=0;
else
    for j=1:length(Gastro)
        if Gastro(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Gastro(j).name;
            all_selected_chemicals_ID(tt,1) =Gastro(j).ID;
            SUS_MAT(i,18)=Gastro(j).score;
            flag=1;
            break
            health_counter=health_counter+1;
        end
    end
    if flag ==0
        SUS_MAT(i,18)=0;
    end
```

```
    flag=0;
end
%%%%%Musculoskeletal_toxicant    skelet
if isempty(skelet)
    SUS_MAT(i,19)=0;
else
    for j=1:length(skelet)
        if skelet(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =skelet(j).name;
            all_selected_chemicals_ID(tt,1) =skelet(j).ID;
            SUS_MAT(i,19)=skelet(j).score;
            flag=1;
            break   %
            health_counter=health_counter+1;
        end
    end
    if flag ==0
        SUS_MAT(i,19)=0;
    end
    flag=0;
end
%%%%%Skin_sense_organ_toxicant   Skin
if isempty(Skin)
    SUS_MAT(i,20)=0;
else
    for j=1:length(Skin)
        if Skin(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Skin(j).name;
            all_selected_chemicals_ID(tt,1) =Skin(j).ID;
            SUS_MAT(i,20)=Skin(j).score;
            flag=1;
            break   %
            health_counter=health_counter+1;
        end
    end
    if flag ==0
        SUS_MAT(i,20)=0;
    end
    flag=0;
end
if health_counter>=1
    OFsCount(1,9)=1;
    %NumOFs=NumOFs+1;
end
%++++++++++++++++++++++++++++++++++++++++++++++++++++safety
%%%Flash point  Flash
if isempty(Flash)
    SUS_MAT(i,21)=0;
else
```

```
    for j=1:length(Flash)
        if Flash(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =Flash(j).name;
            all_selected_chemicals_ID(tt,1) =Flash(j).ID;
            if isnan(Flash(j).point)
                SUS_MAT(i,21)=1000;
            else
                SUS_MAT(i,21)=Flash(j).point;
                OFsCount(1,10)=1;
            end
            flag=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,21)=0;
    end
    flag=0;
end
%%%%%%LEL_UEL    LEL_UEL
if isempty(LEL_UEL)
    SUS_MAT(i,22)=0;
else
    for j=1:length(LEL_UEL)
        if LEL_UEL(j).ID==i
            tt=tt+1;
            all_selected_chemicals(tt,1) =LEL_UEL(j).name;
            all_selected_chemicals_ID(tt,1) =LEL_UEL(j).ID;
            if isnan(Flash(j).point)
                SUS_MAT(i,22)=0;
            else
                SUS_MAT(i,22)=LEL_UEL(j).difference;
                OFsCount(1,11)=1;
            end
            flag=1;
            break
        end
    end
    if flag ==0
        SUS_MAT(i,22)=0;
    end
    flag=0;
end
%%%%%==========TLV TLV
if isempty(TLV)
    SUS_MAT(i,23)=0;
else
    for j=1:length(TLV)
        if TLV(j).ID==i
            tt=tt+1;
```

```
                all_selected_chemicals(tt,1) =TLV(j).name;
                all_selected_chemicals_ID(tt,1) =v(j).ID;
                if isnan(Flash(j).point)
                   SUS_MAT(i,23)=20000;
                else
                   SUS_MAT(i,23)=TLV(j).num;
                   OFsCount(1,12)=1;
                end

                flag=1;
                break
             end
          end
          if flag ==0
             SUS_MAT(i,23)=0;
          end
          flag=0;
       end

    end
    for p=1:9        if OFsCount(1,p)>=1
          NumOFs=NumOFs+1;
       end
    end
    NumOFs=NumOFs+1;

    %crossive index
    material_list = {'Carbon steel','Stainless steel','Better material than tow above',};
    [crossive,p] = listdlg('PromptString','Select the best material used in
process','SelectionMode','single','ListSize',[300 300],'ListString',material_list);

    safetyflag=0;
    component_listoption = questdlg('Is there any difference between component involving in
different process alternative?');

    Safetystructure = questdlg('Is there more than one alternative for process structure with
availble safety data?');
    if strcmpi(Safetystructure,'Yes')
       % process structure safety
       process_structure_list = {'Recommended','Sound engineering practice','No data or
neutral','Probably unsafe','Minor accidents','Major accidents',};
       [processstructure,o] = listdlg('PromptString','Select safety level of the
process','SelectionMode','single','ListSize',[200 300],'ListString',process_structure_list);
    else
       processstructure=1; % just because I have it in the save argument
    end


else
```

```
   crossive =1; processstructure=
1;all_selected_chemicals=[];processstructure=1;all_selected_chemicals_ID=[];component_listopt
ion='No';Safetystructure ='No';

end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%economy and water
existance

Reactionalternatives = questdlg('Is there more than one alternative for the reaction path with
different component?');
switch Reactionalternatives
   case 'Yes'
      NumOFs=NumOFs+1;   % atom economy and also safety index
   case 'No'
      NumOFs=NumOFs;
   case 'Cancel'
      NumOFs=NumOFs;
   otherwise
      NumOFs=NumOFs;
end
water_existance = questdlg('Is any water involved in process?');
switch Safetystructure
   case 'Yes'
      NumOFs=NumOFs+1;
   case 'No'
      NumOFs=NumOFs;
   case 'Cancel'
      NumOFs=NumOFs;
   otherwise
      NumOFs=NumOFs;
end
save('process_chemical_info','SUS_MAT','crossive','processstructure','all_selected_chemicals','all
_selected_chemicals_ID','Reactionalternatives','component_listoption','Safetystructure','water_exi
stance','Number_total_diffrent_chemi','health_social_environmenal');
```

VITA

MAZDAK SHOKRIAN

Candidate for the Degree of

Doctor of Philosophy

Thesis: DEVELOPMENT OF A MULTI CRITERIA PROCESS DESIGN FRAMEWORK FOR SUSTAINABLE DESIGN AND SYNTHESIS OF PROCESSES

Major Field: CHEMICAL ENGINEERING/PROCESS DESIGN

Biographical:

Education:

Completed the requirements for the Doctor of Philosophy in chemical engineering/process design at Oklahoma State University, Stillwater, Oklahoma in December, 2012.

Completed the requirements for the Bachelor of Science in chemical engineering at Iran University of Science and Technology, Tehran, Tehran/IRAN in 2005.

Experience:

Technical Manager, Process Department Computer Aided Process Engineering Company (CAPE)	*2007-2009* *Tehran, Iran*
Engineer, Process Department Computer Aided Process Engineering Company (CAPE)	*2005-2007* *Tehran, Iran*
Part Time Engineering Consultant Kan Investment Company	*2005-2007* *Tehran, Iran*

Professional Memberships:

Member of International Society of Automation (ISA)	2011-Present