New Multi-Objective Optimization Techniques and their Application to Complex Chemical Engineering Problems

by

Allan Vandervoort

A thesis submitted to the Faculty of Graduate and Post Doctoral Studies in partial fulfillment of the requirements for the degree of

Masters of Applied Science

In

Department of Chemical Engineering
Faculty of Engineering
University of Ottawa

January 2011

Acknowledgments

I would like to thank my supervisors Dr. Jules Thibault and Dr. Yash Gupta for their guidance and support throughout all stages of this research.

Statement of Contributions of Collaborators

I hereby declare that I am the sole author of this thesis, and the sole author of the software programs used for simulations in this research. No part of this work has been submitted or accepted for any other degree.

Dr. Jules Thibault and Dr. Yash Gupta supervised this thesis. Both supervisors provided continual guidance throughout this work and made editorial comments and corrections to the written work presented. The responsibilities of the author, Allan Vandervoort, in order to fulfill the requirements of this thesis were as follows.

1. To conduct research in the area of multi-objective optimization in order to determine commonly utilized multi-objective optimization techniques, and how they could be improved.
2. To develop new techniques for approximating the Pareto domain.
3. To apply the new techniques to the optimization of complex chemical engineering problems.
4. To produce four papers for publication based on research performed.
5. To produce a written thesis in partial fulfillment of the requirements for obtaining a Masters in Applied Science.

Signature: _____

Date: _____

Abstract

In this study, two new Multi-Objective Optimization (MOO) techniques are developed. The two new techniques, the Objective-Based Gradient Algorithm (OBGA) and the Principal Component Grid Algorithm (PCGA), were developed with the goals of improving the accuracy and efficiency of the Pareto domain approximation relative to current MOO techniques. Both methods were compared to current MOO techniques using several test problems. It was found that both the OBGA and PCGA systematically produced a more accurate Pareto domain than current MOO techniques used for comparison, for all problems studied. The OBGA requires less computation time than the current MOO methods for relatively simple problems whereas for more complex objective functions, the computation time was larger. On the other hand, the efficiency of the PCGA was higher than the current MOO techniques for all problems tested.

The new techniques were also applied to complex chemical engineering problems. The OBGA was applied to an industrial reactor producing ethylene oxide from ethylene. The optimization varied four of the reactor input parameters, and the selectivity, productivity and a safety factor related to the presence of oxygen in the reactor were maximized. From the optimization results, recommendations were made based on the ideal reactor operating conditions, and the control of key reactor parameters. The PCGA was applied to a PI controller model to develop new tuning methods based on the Pareto domain. The developed controller tuning methods were compared to several previously developed controller correlations. It was found that all previously developed controller correlations showed equal or worse performance than that based on the Pareto domain. The tuning methods were applied to a fourth order process and a process with a disturbance, and demonstrated excellent performance.

Résumé

Dans cette étude, deux nouvelles techniques d'optimisation multicritère (OMC) ont été développées. Ces deux nouvelles méthodes, l'algorithme basé sur le gradient des fonctions objectives (OBGFO) et l'algorithme de la grille des composantes principales (AGCP), ont été développées dans le but d'améliorer la précision et l'efficacité dans la détermination du domaine de Pareto comparée aux techniques OMC actuelles. Les deux méthodes ont été comparées aux techniques d'OMC actuelles pour plusieurs problèmes. Les résultats ont montré que l'OBGFO et l'AGCP produisent systématiquement un domaine de Pareto plus précis que les techniques d'OMC actuelles pour tous les problèmes étudiés. L'OBGFO exige un temps de calcul plus faible que les autres méthodes d'OMC pour des problèmes relativement simples tandis que pour les fonctions d'objectives plus complexes, le temps de calcul est supérieur. D'autre part, l'efficacité de l'AGCP était meilleure que les techniques d'OMC actuelles pour tous les problèmes étudiés.

Les nouvelles techniques ont aussi été utilisées pour des problèmes reliés au génie chimique. L'OBGFO a été utilisé pour optimiser un réacteur industriel servant à la production de l'oxyde d'éthylène. Quatre paramètres d'entrée du réacteur ont été variés et la sélectivité, la productivité et un facteur de sécurité lié à la présence d'oxygène dans le réacteur ont été maximisés. Les résultats d'optimisation ont permis de faire certaines recommandations sur les conditions d'opération du réacteur et sur le contrôle des paramètres clés du réacteur. L'AGCP a été utilisé pour développer de nouvelles méthodes d'ajustement d'un contrôleur PI fondées sur le domaine de Pareto. Les méthodes d'ajustement développées du contrôleur ont été comparées à plusieurs corrélations développées par d'autres chercheurs. Il a été montré que toutes les méthodes développées précédemment étaient égales ou pires que celles fondées sur le domaine de Pareto. Les méthodes d'ajustement ont été appliquées au contrôle d'un procédé de quatrième ordre et un procédé soumis à une perturbation. Les résultats ont montré une excellente performance.

Table of Contents

List of Figures and Tables

Chapter 1: An Objective-Based Gradient Method for Locating the Pareto Domain

Chapter 2: Multi-Objective Optimization of an Ethylene Oxide Reactor

Chapter 3: A Principal Component Grid Algorithm for Approximating the Pareto Domain

Chapter 4: New PI Controller Tuning Methods using Multi-Objective Optimization

Introduction

1.0 Multi-Objective Optimization

In complex industrial processes, several important process variables need to be optimized simultaneously to ensure that the process is operating in an efficient and cost-effective manner. This is especially important when some of the objectives are conflicting, such as environmental and economic concerns in a chemical process. The multiple objectives are often combined into a unique weighted objective function. This method allows finding an optimum value but does not reveal information about the trade-off between each objective function. In addition, the weighting must be chosen carefully to ensure finding a solution that satisfies the expectation of the decision-maker. Moreover, the amalgamation of all objective functions into a unique objective function may lead to a solution that is a local optimum. For this reason further testing may be required to locate the global optimum, which may result in greater computation time (Deb, 2001).

In recent years, to circumvent the limitations associated with the weighted objective methods, multi-objective optimization (MOO) techniques have been developed which allow for several objectives to be optimized simultaneously (Deb, 2001). In MOO the best operating region, known as the Pareto domain, is first identified. Several studies have proposed algorithms for determining the Pareto domain (Deb et al., 2002; Halsall-Whitney et al., 2006; Poloni et al., 2000; Fonteix et al., 2004; Viennet et al., 1995). In this study, two new methods for approximating the Pareto domain were developed with the goals of improving the efficiency and accuracy of the Pareto domain approximation relative to current MOO techniques. A brief description of current MOO techniques that were considered in this study is given below.

2.0 Current Methods for Approximating the Pareto domain

2.1 Grid Search Approach

The grid search approach (GSA) is a technique used to approximate the Pareto domain that involves the construction of a grid in the input space of an optimization problem. The objective functions are then calculated at the grid points, and the best points are chosen. The search begins from a coarse grid, and is refined at each iteration. Of the MOO techniques used to approximate the Pareto domain, a grid-based search is one of the simplest and easiest

to implement. The GSA also ensures a Pareto domain which evenly and fully spans the input space. However, the GSA can lead to an approximation of the Pareto domain with limited accuracy, requiring high computation time (Halsall-Whitney et al., 2006)

2.2 Genetic Algorithms

Genetic algorithms are a class of optimization techniques that are inspired by natural evolution. These algorithms are initialized by a random set of points, normally referred to as the initial population of points. The initial population and subsequent generations are progressively improved by generating new populations based on random variation or combination of points from the previous population (Salomon, 1998). Two specific genetic algorithms were considered in this study. These were the Dual Population Evolutionary Algorithm (DPEA) developed by Halsall-Whitney et al. (2006), and the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) developed by Deb et al. (2002). Genetic algorithms are very commonly utilized in optimization studies, but have the limitation of being complex, often requiring high computation time to approximate the Pareto domain.

2.3 Gradient-Based Algorithms

Gradient-based algorithms relate to steepest descent optimization for a minimization problem, or steepest ascent for a maximization problem. An initial population of points can be moved closer toward the Pareto domain, by determining the direction of steepest descent, and making a change in the input variables such that an improvement in the objective functions is realized. These algorithms move solutions towards the Pareto domain, but it is not known explicitly in what direction the output functions will move. For example, if gradient descent is performed on an initial population with very evenly spaced solutions, the even spacing may not be maintained after gradient descent, as each solution may not move in the same direction toward the Pareto domain.

3.0 Objectives of this Research

The objective of this research was to develop new methods for approximating the Pareto domain. Two new methods were developed with the goals of improving the accuracy and efficiency relative to previously developed MOO techniques. Several test problems were studied to evaluate the performance of the developed algorithms, and the results were compared to current methods used to approximate the Pareto domain. Both developed

optimization methods were also applied to complex chemical engineering problems, specifically the reactor producing ethylene oxide from ethylene and a PI controller model. A brief description of each paper produced for publication during this research is shown below.

3.1 The Objective-Based Gradient Algorithm

In the first paper, An Objective Based Gradient Algorithm for Approximating the Pareto Domain, the first MOO technique was developed. Many current methods for approximating the Pareto domain involve calculations which are based on the input functions of an optimization problem. The Objective-Based Gradient Algorithm (OBGA) considers the objective functions for an optimization problem directly when approximating the Pareto domain. The accuracy and efficiency of the OBGA in approximating the Pareto domain was compared to the standard MOO techniques NSGA-II and DPEA, using several test problems.

3.2 The Ethylene Oxide Reactor

In the second paper, Multi-Objective Optimization of an Industrial Ethylene Oxide Reactor, the OBGA was applied to an industrial reactor producing ethylene oxide from ethylene. This reactor has been studied by several authors but in previous studies the Pareto domain was not approximated, and information about the multiple optimum values for the reactor parameters and the trade-off between the various objectives was not revealed. In this study, the optimization problem consisted of four reactor input parameters and three objective functions (the selectivity, the productivity and a safety factor related to the presence of oxygen in the reactor) which were all maximized. After approximating the Pareto domain, the numerous solutions were ranked using the Net-Flow method. Finally, recommendations about the operation of the ethylene oxide reactor and the control of key reactor parameters were made based on the highest ranked solutions in the Pareto domain.

3.3 The Principal Component Grid Algorithm

In the third paper, A Principal Component Grid Algorithm for Approximating the Pareto Domain, a second method for approximating the Pareto domain was developed. The GSA is a simple method for approximating the Pareto domain that is easy to implement, but redundant points generated at each iteration can lead to limited accuracy and high computation time. The Principal Component Grid Algorithm (PCGA) combines the Principal component analysis with a GSA, with the goals of enhancing the accuracy and the efficiency

of the approximation of the Pareto domain by reducing the grid search region. The accuracy and efficiency of the PCGA in approximating the Pareto domain was compared to the standard MOO technique NSGA-II and the GSA using several test problems.

3.4 PI Controller Tuning Methods

In the fourth paper, New PI Controller Tuning Methods using Multi-Objective Optimization, the PCGA was applied to the development of new PI controller tuning methods. Although many PI controller tuning methods have been developed, including methods based on Multi-Objective Optimization, no current method allows the decision maker to fully integrate the trade-off associated with each performance objective when choosing optimum controller parameters. The PI controller methods developed in this study involve approximating the Pareto domain associated with the minimization of three performance criteria, the ITAE, ISDU and settling time. The new PI controller tuning methods were developed with the goal of improving the decision maker's understanding of the trade-off associated with each objective. To evaluate the proposed tuning methods they were compared to several current controller correlations, and applied to multiple process models.

4.0 References

Deb K. Multi-objective optimization using Evolutionary Algorithms. England: John Wiley & Sons, Ltd; 2001.

Deb K, Pratap A, Agarwal S, Meyarivan T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Trans. On Evol. Comput. 2002; 6: 182-97.

Fonteix C, Massebeuf S, Pla F, Kiss L. Multicriteria optimization of an emulsion polymerization process. Eur. J. Oper. Res. 2004; 153: 350–9.

Halsall-Whitney H, Thibault J. Multi-objective optimization for chemical processes and controller design: Approximating and classifying the Pareto domain. Comput. Chem. Eng 2006; 30: 1155-68.

Poloni C, Giurgevich A, Onesti L, Pediroda V. Hybridization of a multi-objective genetic algorithm, a neural network, and a classical optimizer for a complex design problem in fluid dynamics. Comput. Method in Appl. Math. 2000; 186: 403-20.

Salomon R. Evolutionary Algorithms and Gradient Search: Similarities and Differences. IEEE Trans. On Evol. Comput. 1998; 2: 60; 45-55.

Viennet R, Fonteix C, Marc I. Multicriteria optimization using a genetic algorithm for determining a Pareto set. Int. J. Syst. Sci. 1996; 27: 255-60.

Chapter 1

An Objective-Based Gradient Method for Locating the Pareto Domain

Allan Vandervoort, Jules Thibault* and Yash Gupta

Abstract

In this paper, an objective-based gradient multi-objective optimization (MOO) technique, the Objective-Based Gradient Algorithm (OBGA), is proposed with the goal of defining the Pareto domain more accurately and efficiently than current MOO techniques. The performance of the OBGA in locating the Pareto domain was evaluated in terms of accuracy, computation time and number of objective function calls, and compared to two current MOO algorithms: Dual Population Evolutionary Algorithm (DPEA) and Non-Dominated Sorting Genetic Algorithm II (NSGA-II), using four test problems. For all test problems, the OBGA systematically produced a more accurate Pareto domain than DPEA and NSGA-II. With the adequate selection of the OBGA parameters, computation time required for the OBGA can be lower than that required for DPEA and NSGA-II. Results clearly show that the OBGA is a very effective and efficient algorithm for locating the Pareto domain.

Keywords: Pareto domain, Multi-Objective Optimization, Gradient Method

Journal Publication: Journal of Chemistry and Chemical Engineering
Publisher: David Publishing
Publication Status: Submitted

1.0 Introduction

In complex industrial processes, several important process variables need to be optimized simultaneously to ensure that the process is operating in an efficient and cost-effective manner. This is especially important when the objectives are conflicting, such as environmental and economic concerns in a chemical process. The multiple objectives are often combined into a unique weighted objective function. This method allows finding an optimum value but does not reveal information about the trade-off between each objective function. In addition, the weighting must be chosen carefully to ensure finding a solution that satisfies the expectation of the decision-maker. Moreover, the amalgamation of all objective functions into a unique objective function may lead to a solution that is a local optimum. For this reason further testing may be required to locate the global optimum, which may result in greater computation time (Deb, 2001).

In recent years, to circumvent the limitations associated with the weighted objective methods, multi-objective optimization (MOO) techniques have been developed which allow for several objectives to be optimized simultaneously (Deb, 2001). In MOO the best operating region, known as the Pareto domain, is identified. Several studies have proposed algorithms for determining the Pareto domain (Deb et al., 2002; Halsall-Whitney et al., 2006; Poloni et al., 2000; Fonteix et al., 2004; Viennet et al., 1995). In this study, a new method for locating the Pareto domain is developed and tested using a series of problems. Two of the most common methods for approximating the Pareto domain are genetic algorithms and gradient-based algorithms. In both methods, calculations performed to progressively reach values of the objective functions (or output space) to adequately approximate the Pareto domain, are based on the input space. In the proposed method, calculations are based directly on the output space (objective criteria). The proposed method has been developed with the goal of defining the Pareto domain more accurately and efficiently than current MOO techniques.

2.0 Current Algorithms for Determining the Pareto Domain

2.1 The Pareto Domain

Prior to discussing current methods used to circumscribe the Pareto domain, a description of the Pareto domain and the concept of dominance are briefly reviewed. Before the decision-maker chooses a good compromise solution by considering tradeoffs between the competing

criteria, the search domain can be significantly reduced by only considering solutions that would potentially be candidates in the selection of the optimal solution. The set of retained solutions, called the Pareto domain, is obtained based on the concept of dominance.

In Figure 1, the concept of dominance and the Pareto domain are described graphically using a simple illustrative example with two output functions, f_1 and f_2, which depend on two input variables, x_1 and x_2. In this example, both functions are to be maximized. Four points are used for this illustration. Point A has the lowest values for both output functions. It can therefore be stated that point A is dominated by points B, C and D and, under no circumstances, point A will be considered optimal. Although point B dominates point A, point B has lower output function values than points C and D. Therefore point B is also dominated by points C and D. When points C and D are compared to each other, both points are higher in one output function value, and lower in the other. Therefore points C and D are said to be non-dominated points. In this example, points C and D would belong to the Pareto domain if no additional points with higher values for both output functions were generated.

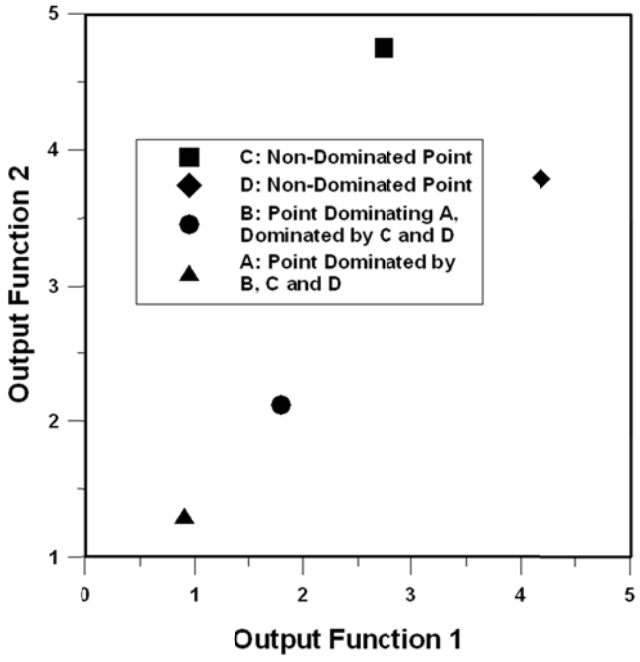

Figure: 1 Illustration of the concept of dominance to define the Pareto domain.

For a general definition of dominance, consider two points, P_1 and P_2, comprised of n input variables (x_1, x_2, x_3 … x_n) and m output (objective criteria) values (f_1, f_2, f_3 … f_m). For a point P_1 to dominate a point P_2, the following two conditions must hold (Deb, 2001):

- None of the criteria values, f_1 to f_m, for P_1 are worse than the corresponding criteria values, f_1 to f_m, for P_2. For example, if all output criteria values are to be maximized then no criteria values in P_1 can be smaller than the corresponding criteria values for P_2.
- At least one objective criterion for P_1 must be better than the corresponding objective criterion for P_2.

If P_1 dominates P_2 then P_2 is a dominated point and if P_1 does not dominate P_2 and P_2 does not dominate P_1, than both are non-dominated points with respect to each other.

For a given optimization problem, the Pareto domain is the region within the domain of all feasible solutions that contains only non-dominated points. This means that all points outside of the Pareto domain are dominated points and therefore worse for all criteria than a particular point in the Pareto domain.

2.2 Genetic Algorithms

Genetic algorithms are a class of optimization techniques that are inspired by natural evolution. Specific details vary greatly between algorithms, but the basic concept is similar. These algorithms are initialized by a random set of points, normally referred to as the initial population of points. The initial population and subsequent generations are progressively improved by generating new populations based on random variation or combination of points from the previous population (Salomon, 1998). Two specific genetic algorithms were used in this study and compared with the proposed algorithm. These algorithms are briefly described below.

2.21 Dual Population Evolutionary Algorithm

The Dual Population Evolutionary Algorithm (DPEA) used in this study was developed by Halsall-Whitney et al. (2006). This method is a modified version of the diploid genetic algorithm (Fonteix et al., 2004) used to approximate the Pareto domain. This algorithm can be described using the example discussed in Section 2.1. First, a large number of points are randomly generated for the input variables, x_1 and x_2, within their feasible regions. Next, the

corresponding output function values, f_1 and f_2, are calculated. All points are compared to each other in pairs to determine the number of times a given solution is dominated by another. All points in the current population are then sorted based on the number of times they were dominated, so that a point that is dominated by many other points is considered worse than a point that is not dominated or dominated by only a few other points. All non-dominated points and a fraction of the least dominated points are retained and used to generate new points to replace the dominated points that were discarded. This ensures that the number of points from one generation to the next remains the same. Retaining a fraction of the dominated points provides the opportunity of extending the search area and better defining the Pareto domain. Each new point is generated by randomly selecting two points, one non-dominated and one dominated, from the retained set of points, performing a random linear interpolation between the input variables of the two selected points and calculating the corresponding objective function values. All of the points in the new generation are again sorted based on dominance and the above procedure is repeated until a predetermined number of non-dominated points are obtained. A more detailed description of this algorithm can be found in Halsall-Whitney et al. (2006).

The DPEA algorithm has the advantage of being straightforward and easy to implement. During the development of this study it was found that DPEA initially converges quickly to a relatively accurate approximation of the Pareto domain. Despite these advantages, DPEA does not consider the spatial distribution of the numerous solutions and the resulting Pareto domain is limited to the points found in the initial generation of the population and their interpolation. DPEA also demonstrates slow convergence when the percentage of dominated points becomes relatively low and only a few non-dominated points can be identified at each subsequent iteration.

2.22 Non-Dominated Sorting Algorithm II

Non-Dominated Sorting Genetic Algorithm II (NSGA-II) is a genetic algorithm developed by Deb et al. (2002) that has been commonly utilized for a wide range of MOO problems. Examples of MOO procedures that have been performed using NSGA-II include a shell and tube heat exchanger (Agarwal et al., 2008), a multi-product batch chemical process (Mokeddem et al., 2009), a styrene manufacturing process (Tarafder et al., 2005), a control

scheme for a biochemical process (Logist et al., 2009) and environmentaly conscious design of a chemical process (Li et al., 2009). NSGA-II also begins with an initial random population, but further generations are formed based on natural evolution, involving both crossover and random mutation. Points that are retained in a population are chosen based on dominance as well as a factor known as the crowding distance, to ensure a more uniform distribution of the objective function values within the Pareto domain. The algorithm can proceed for a given number of iterations, or until a predetermined number of non-dominated points are found. A detailed description of this algorithm can be found in Deb et al. (1998, 2002).

NSGA-II favors population members with the largest crowding distance. This can aid in producing a well distributed final Pareto domain. Also, since NSGA-II involves mutation from parent generations the accuracy of the final Pareto domain is not limited by points in the initial population. One of NSGA-II's disadvantages is the relatively high number of iterations to initially find non-dominated points, especially if the feasible regions for the input variables are large (Lalonde, 2009). This can lead to high computational time in locating the Pareto domain. NSGA-II is also more complex to implement than DPEA.

2.3 Gradient-Based Algorithms

Gradient-based algorithms offer an alternative to genetic algorithms. Gradient-based algorithms relate to steepest descent optimization for a minimization problem, or steepest ascent for a maximization problem. For a minimization problem with *m* output variables and *n* input variables, the algorithm proceeds as follows:

1. First a point is chosen in the input space. It is desired to move this point such that the corresponding objective function values move toward the Pareto domain.
2. By calculating the derivatives of each output variable in terms of each input variable, the direction of steepest descent for each objective can be calculated.
3. The input variables are then modified in small step changes such that the associated objective functions move towards the Pareto Domain.
4. The process is then repeated, gradually moving the specific point towards the Pareto domain.

Mathematically, this algorithm can be represented with Equation (1) for a system of n input variables and m output variables. Equation (1) evaluates the change in each input variable based on the steepest descent for objective i.

$$\begin{aligned} dx_1 &= -\eta \frac{\partial f_i}{\partial x_1} \\ dx_2 &= -\eta \frac{\partial f_i}{\partial x_2} \qquad or \quad d\underline{x} = -\eta \nabla f_i(\underline{x}) \\ &\vdots \\ dx_n &= -\eta \frac{\partial f_i}{\partial x_n} \end{aligned} \tag{1}$$

Where $d\underline{x}$ is the vector of changes dx_j for each of the n input variables, η is the step adjustment factor, $\nabla f_i(\underline{x})$ is the gradient vector of partial derivatives with respect to each input variable for objective i ($-\nabla f_i(\underline{x})$ corresponds to the direction of steepest descent for objective i) (Salomon, 1998).

Gradient-based MOO techniques can involve moving many random initial points towards the Pareto domain using the steepest descent (Brown et al., 2005), incorporating gradient information into genetic algorithms (Arnold et al., 2007) or using genetic algorithms to form an initial population prior to implementing the gradient descent (Bosman et al., 2005). It should be noted that the direction of steepest descent can be different for each of the m objectives, and an overall direction of descent must be chosen. The estimation of the overall direction of the steepest descent varies with the specific algorithm. For instance, Bosman et al. (2005) estimated the steepest descent direction via a linear combination of the gradient direction for each objective function. These algorithms move points towards the Pareto domain, but it is not known explicitly in what direction the output functions will move towards the Pareto domain. For example, if gradient descent is performed on an initial population with very evenly spaced solutions, the even spacing may not be maintained after gradient descent, as each solution may not move in the same direction toward the Pareto domain.

3.0 The Proposed Method: the Objective-Based Gradient Algorithm

In this paper, a new gradient-based method, the Objective-Based Gradient Algorithm (OBGA), is proposed. Since in the output (objective criteria) space, the general direction in which the points should be moved is known, the decisions regarding the amount and direction of movement of points are made in the output space. The partial derivatives of each objective function with respect to each input variable for a given point, allow the calculation of the required change in the input variables to produce the desired change in the output variables. Instead of making changes in the input space to move the objective functions toward the Pareto domain, points can be moved closer to the Pareto domain by making calculations based directly on the output space.

3.1 Algorithm Description

For the OBGA, the partial derivatives for a specific point in the initial population are first calculated. For many engineering problems, analytical derivatives are not available and one must resort to numerical derivatives as expressed by Equation (2).

$$\frac{df_i}{dx_j} = \frac{f_i(x_j + \Delta x_j) - f_i(x_j)}{\Delta x_j} \tag{2}$$

f_i represents the value of output i ($\in[1,m]$) of the given point, and x_j represents the corresponding value of input j ($\in[1,n]$). These derivatives are calculated for each input j and each output i. The overall change in each output function (from df_1 to df_m), based on the change in all input variables (dx_1 to dx_n), can be calculated using the linear approximation shown in Equation (3).

$$\begin{aligned}
df_1 &= \frac{\partial f_1}{\partial x_1}dx_1 + \frac{\partial f_1}{\partial x_2}dx_2 + \frac{\partial f_1}{\partial x_3}dx_3 + \cdots + \frac{\partial f_1}{\partial x_n}dx_n \\
df_2 &= \frac{\partial f_2}{\partial x_1}dx_1 + \frac{\partial f_2}{\partial x_2}dx_2 + \frac{\partial f_2}{\partial x_3}dx_3 + \cdots + \frac{\partial f_2}{\partial x_n}dx_n \\
&\vdots \\
df_m &= \frac{\partial f_m}{\partial x_1}dx_1 + \frac{\partial f_m}{\partial x_2}dx_2 + \frac{\partial f_m}{\partial x_3}dx_3 + \cdots + \frac{\partial f_m}{\partial x_n}dx_n
\end{aligned} \tag{3}$$

where each element in matrix $\underline{\underline{D}}$ (the Jacobian Matrix) is $\frac{\partial f_i}{\partial x_j}$ and represents the partial derivatives of the objective function *i* relative to input variable *j*, at the point of interest. Element df_i of vector $\underline{df}$ represents the desired change in the output function *i* and element dx_j of vector $\underline{dx}$ represents the corresponding change in the input variable *j*.

Since $\underline{dx}$ is to be calculated for specified values of $\underline{df}$, Equation (3) is rewritten in the following form for the OBGA:

$$\underline{dx} = \underline{\underline{D}}^{-1} \, \underline{df} \tag{4}$$

if the number of inputs is equal to the number of outputs or as

$$\underline{dx} = \left(\underline{\underline{D}}^T \underline{\underline{D}}\right)^{-1} \underline{\underline{D}}^T \underline{df} \tag{5}$$

if the number of inputs is not equal to the number of outputs (Equation 5 is the pseudo inverse). The definition of matrix $\underline{\underline{D}}$ and vectors $\underline{dx}$ and $\underline{df}$ in Equation (3) also apply to Equations (4) and (5).

The OBGA provides better control in moving solutions toward the Pareto domain when compared to standard gradient-based techniques. For example, if the OBGA is performed on an initial population with very evenly spaced solutions, the accuracy of the initial population can be increased while the even spacing is maintained by moving each solution toward the Pareto domain in the same direction. The OBGA also allows determining the final accuracy of the Pareto domain approximation, which is often not possible with standard gradient-based techniques, or genetic algorithms. After performing OBGA, it can be stated that the Pareto domain has been approximated within the direction and spacing of the final $\underline{df}$ values chosen.

3.2 Graphical Illustration

A graphical illustration showing a typical progression of the OBGA for a problem with 2 input variables and 2 output functions is shown in Figure 2. The calculation begins from a selected point and the desired changes in df_1 and df_2 are specified. The required changes in the input values, dx_1 and dx_2, are then calculated using Equation (4), which takes the specific form of Equation (6) for this two-by-two problem.

$$\begin{bmatrix} dx_1 \\ dx_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}^{-1} \begin{bmatrix} df_1 \\ df_2 \end{bmatrix} \tag{6}$$

New input values, x_1 and x_2, and the corresponding output function values, f_1 and f_2, are then calculated. If the chosen values of df_1 and df_2 correspond to a feasible point which does not exceed the Pareto domain, the new point obtained will dominate the original point. This calculation procedure is repeated each time a new point is obtained until the resulting point reaches a location that is approaching the Pareto domain. Beyond this point, the next calculation will generate a dominated point. This indicates that the Pareto domain has been reached within the chosen values of df_1 and df_2. The values of df_1 and df_2 are known as the tolerance in this study. Figures 2 and 3 show respectively the progression of the output variables and the associated input variables toward the Pareto domain. The arrows in the figure indicate the direction of movement of variables during the calculation procedure. In this simple illustrative example, the search was ended after obtaining a dominated point. In practice, to approximate the Pareto domain within the desired accuracy, once a dominated point is found the values of df_1 and df_2 are decreased and the process is continued until predetermined minimum values of $\underline{df}$ have been reached.

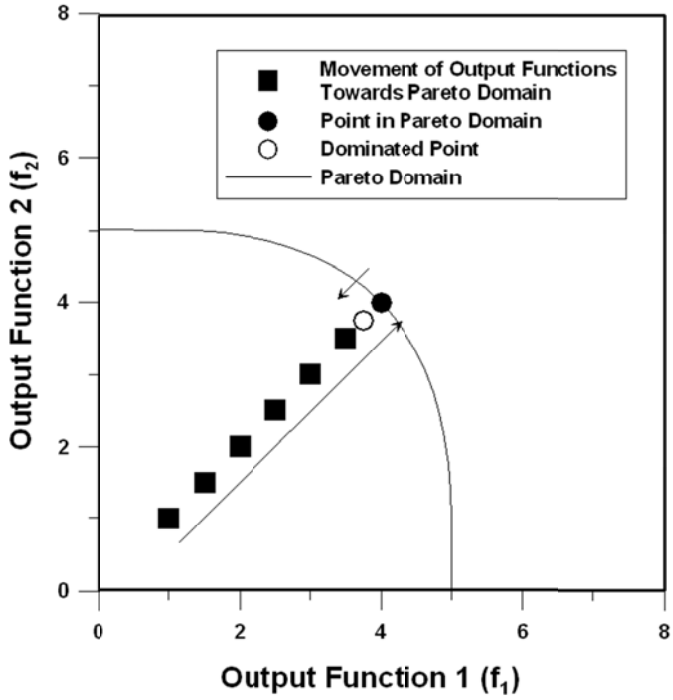

Figure 2: Illustration of a typical change in two output functions using the OBGA.

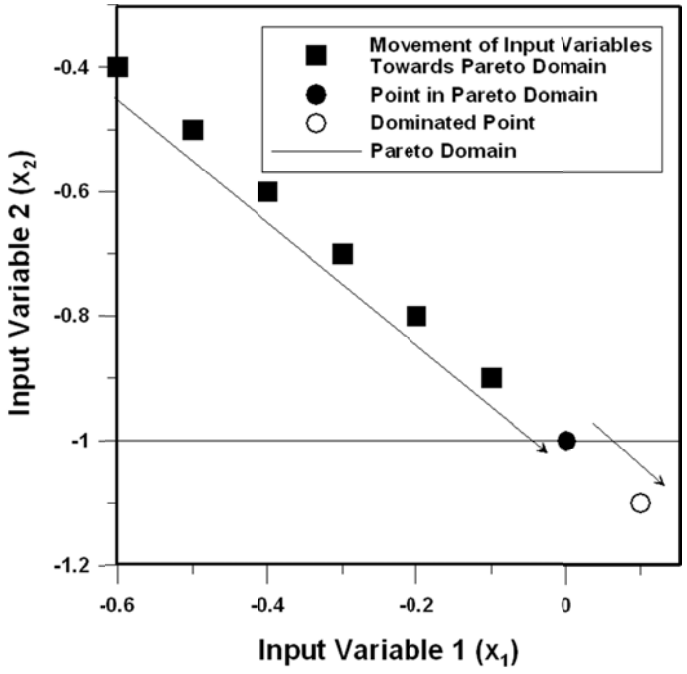

Figure 3: Illustration of the associated changes of the input variables using the OBGA.

3.3 Choice of the Tolerance

In this study all values of df_i for each objective were initially set to a value of 0.1 for a maximization problem, or -0.1 for a minimization problem unless otherwise noted. With these values of df_i the tolerance was said to be equal to 0.1. For the first iteration the procedure described in Section 3.1 is applied to each solution in the initial population for the specified tolerance. This initial tolerance value is relatively large, and for some solutions movement in the specified direction may not be possible. In this situation a dominated point will be immediately generated, and the original point is maintained. The next iteration begins by decreasing the values of each df_i by a factor of 10, corresponding to a tolerance value of 0.01. As the tolerance is decreased the occurrence of solutions which cannot be moved toward the Pareto domain is also decreased. The tolerance is decreased by a factor of 10 at each iteration until the desired final tolerance, representing the final accuracy of the approximation of the Pareto domain, is reached.

It should be noted that for many theoretical test problems choosing the same value for each df_i is sufficient, as the magnitude of the range of each objective found in the Pareto domain is similar. For many engineering problems this may not be sufficient, as the magnitude of the objectives may be significantly different, for example if one objective is the concentration of a trace component and one objective is the concentration of a major product in a reaction process. In this case the same procedure as discussed above is followed for the tolerance, but a scaling factor is applied to each df_i such that the value of each df_i is equal to the tolerance multiplied by the scaling factor for each objective. The scaling factor can be easily calculated from the range in each objective in the initial population before the calculation procedure discussed in Section 3.1 is applied.

3.4 Initial Population

An initial set of points is first required to perform the OBGA. The initial population can be a random set of points or a Pareto domain approximated using a more conventional method. In this study, a genetic algorithm was used to provide the initial set of points necessary for the OBGA. During the development of this study it was found that the DPEA is able to converge rapidly to a number of non-dominated solutions that are relatively close to the Pareto domain. On the other hand, the convergence rate decreases significantly with the subsequent number

of iterations. It was also found that NSGA-II shows opposite behavior, converging slowly in the initial steps and more rapidly after many iterations. DPEA was therefore chosen as the initial population for this study, as it reaches an approximation of the Pareto domain suitable for the OBGA more quickly than NSGA-II.

4.0 Test Problems

Several problems were chosen to test the ability of the OBGA to adequately approximate the Pareto domain. To ensure validity across a wide range of optimization problems, test problems were chosen to include a range in the number of input and output variables, and a range of problem types.

4.1 Basic Problem (Problem 1)

The first optimization problem is a simple system comprised of two input variables and two output functions as shown in Equations (7) and (8). Both objective functions are to be minimized (Deb, 2001).

$$f_1(x_1, x_2) = x_1^2 + x_2^2 \tag{7}$$

$$f_2(x_1, x_2) = (x_1 + 2)^2 + x_2^2 \tag{8}$$

$-50 < x_1 < 50$ and $-50 < x_2 < 50$

4.2 Disjointed Pareto Domain (Problem 2)

The second problem, described by Equations (9) and (10), is comprised of 2 input variables and 2 output functions. This problem is more complex as it is a multi-modal problem, consisting of two distinct Pareto regions. This problem also involves input variables at the outer boundary of their predetermined range, and therefore ensures that the optimization procedure correctly accounts for constraints on the input variables. Both output functions are to be minimized (Poloni et al., 2000).

$$f_1(x_1, x_2) = 1 + (A_1 - B_1)^2 + (A_2 - B_2)^2 \tag{9}$$

$$f_2(x_1, x_2) = (x_1 + 3)^2 + (x_2 + 1)^2 \tag{10}$$

where

$$A_1 = 0.5\sin(1) - 2\cos(1) + \sin(2) - 1.5\cos(2) = 0.8736$$
$$A_2 = 1.5\sin(1) - \cos(1) + 2\sin(2) - 0.5\cos(2) = 2.7486$$
$$B_1 = 0.5\sin(x_1) - 2\cos(x_1) + \sin(x_2) - 1.5\cos(x_2)$$
$$B_2 = 1.5\sin(x_1) - \cos(x_1) + 2\sin(x_2) - 0.5\cos(x_2)$$

$\pi < x_1 < -\pi$ and $\pi < x_2 < -\pi$

4.3 Non-square Problem (Problem 3)

The OBGA procedure involves finding the inverse of the partial derivative matrix for a given problem. For non-square problems (i.e. when the number of input functions does not equal the number of output functions) a pseudo-inverse is used. For this reason, a non-square problem was chosen to ensure the optimization algorithm functions correctly for these types of problems. The problem described by Equations (11)-(13) involves three output functions and two input variables. All three output functions are to be minimized (Viennet et al., 1995).

$$f_1(x_1, x_2) = 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \tag{11}$$

$$f_2(x_1, x_2) = \frac{(3x_1 - 2x_2 + 4)^2}{8} + \frac{(x_1 - x_2 + 1)^2}{27} + 15 \tag{12}$$

$$f_3(x_1, x_2) = \frac{1}{x_1^2 + x_2^2 + 1} - 1.1\ \exp(-(x_1^2 + x_2^2)) \tag{13}$$

$-3 < x_1 < 3$ and $-3 < x_2 < 3$

4.4 Production of Gluconic Acid (Problem 4)

The fourth problem involves the integration of differential equations. Since the required computation time for each objective function call is large, the total computation time associated with the objective function calls is important. The problem is concerned with the production of gluconic acid from glucose using the microorganism *Pseudomonas ovalis*. The overall reaction can be described as follows.

Cells + Glucose + Oxygen $\longrightarrow$ More Cells

Glucose + Oxygen $\xrightarrow{\text{Cells}}$ Gluconolactone

Gluconolactone + Water $\longrightarrow$ Gluconic Acid

The dynamic model of this fermentation process, given in Equations (14)-(18), was developed by Ghose and Ghosh (1971) and accounts for the changes in the concentration of cells (X), gluconic acid (P), gluconolactone (l), glucose substrate (S), and dissolved oxygen (C). The parameters for the model are given in Table 1.

$$\frac{dX}{dt} = \mu_m \frac{SC}{k_s C + k_0 S + SC} X \tag{14}$$

$$\frac{dP}{dt} = k_p l \tag{15}$$

$$\frac{dl}{dt} = v_l \frac{S}{k_l + S} X - 0.91 k_p l \tag{16}$$

$$\frac{dS}{dt} = -\frac{1}{y_s} \mu_m \frac{SC}{k_s C + k_0 S + SC} X - 1.011 v_l \frac{S}{k_l + S} X \tag{17}$$

$$\frac{dC}{dt} = K_L a (C^* - C) - \frac{1}{y_0} \mu_m \frac{SC}{k_s C + k_0 S + SC} X - 0.09 v_l \frac{S}{k_l + S} X \tag{18}$$

Table 1: Parameters used in the gluconic acid production model.

Parameter	Value	Unit
μ_m	0.39	h^{-1}
k_s	2.5	g/L
k_0	0.00055	g/L
k_P	0.645	h^{-1}
v_l	8.3	mg/UOD h
K_l	12.8	g/L

Y_s	0.375	UOD/mg
Y_0	0.89	UOD/mg
C*	0.00685	g/L
X_0 (initial cell concentration)	1	UOD/mL
S_0 (initial substrate concentration)	50	g/L

For this model, there exist a large number of input and output variables. In this study, two input variables, the batch time t_B (x_1) and the overall oxygen mass transfer coefficient K_La (x_2), were varied in the optimization, and all other potential input variables remained constant. Two output variables, the productivity P_f/t_B (f_1) and the final concentration of gluconic acid P_f (f_2) were retained (Thibault, 2009) such that the optimization problem is a two-input and two-output system. The lower and upper bounds for the two input variables are $5 < t_B < 15$ h and $50 < K_La < 300$ h^{-1}. In this study, Equations (14)-(18) were integrated using Euler's method, with a time step of 0.001.

5.0 Results and Discussion

To evaluate the performance of the OBGA, Pareto domains obtained with the OBGA were compared to Pareto domains obtained with DPEA and NSGA-II. It is important to compare the performance of OBGA to that of DPEA because DPEA was used as the initial population for the OBGA in this study. A comparison between these two methods therefore clearly demonstrates improvements realized from the addition of the objective-based gradient descent. The Pareto domain obtained by the OBGA was also compared to that obtained by NSGA-II. NSGA-II is a very commonly used algorithm in the literature for MOO problems, and it is therefore important to compare its performance to that of OBGA. Also, it was demonstrated by Lalonde (2009) that the accuracy obtained by NSGA-II is greater than the accuracy obtained by the popular gradient-based algorithm, the normalized-normal constraint method, for optimization problems with bounded input variables (which include the majority of engineering optimization problems). Therefore any improvement in the accuracy obtained

by OBGA relative to NSGA-II will also show improvement in accuracy relative to the normalized-normal constraint method.

It should be noted that due to the variability of the genetic algorithms used in this study, both as a comparison and as an initial population for the OBGA, all results shown are based on an average of six runs for each method. All computer simulations were performed with a 4.5 GHz Intel Pentium 4 processor with 1.49 GB of RAM.

5.1 Effect of the Final Tolerance

In this study, the effect of key parameters used in the OBGA was examined. The first parameter studied was the final tolerance which sets the maximum distance a given point is located from the Pareto domain upon convergence. Problem 2 was used to determine the effect of the tolerance on the OBGA. Problem 2 was used for this comparison as it is multi-modal, with two distinct regions in the Pareto domain. Gradient-based methods have difficulties locating the Pareto domain for multi-modal problems (Lalonde, 2009), but incorporating a genetic algorithm as an initial population allows the OBGA to successfully locate the Pareto domain for these types of problems. The Pareto domains for NSGA-II, DPEA and the OBGA were approximated for Problem 2. The NSGA-II and DPEA procedures were each performed for a population of 125 points, until the population contained only non-dominated points. The tolerance for the OBGA was varied from 0.3 to 10^{-7}. It should be noted that for Problem 2 the magnitude of the range of the two objective functions were similar, and scaling was therefore not necessary as discussed in Section 3.3. Initially the Pareto domains were inspected visually to estimate the accuracy of each of the three algorithms. An example of the Pareto domain for this problem is shown in Figures 4 and 5 for a final tolerance value of 10^{-4} for the OBGA, and a population size of 125 for the DPEA.

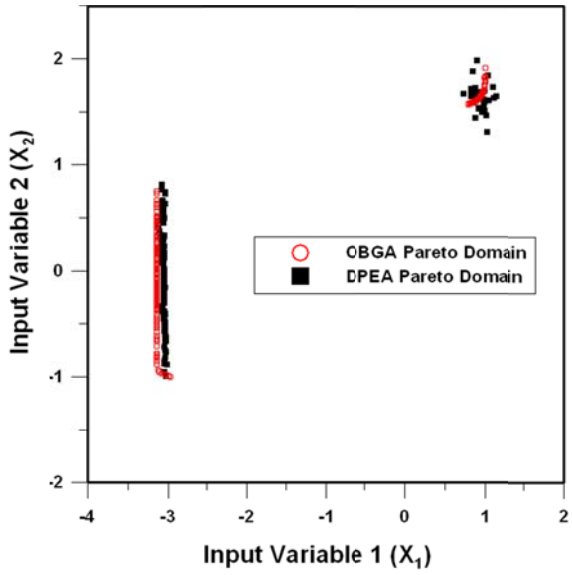

Figure 4: Input space for both the DPEA and OBGA Pareto domains for Problem 2.

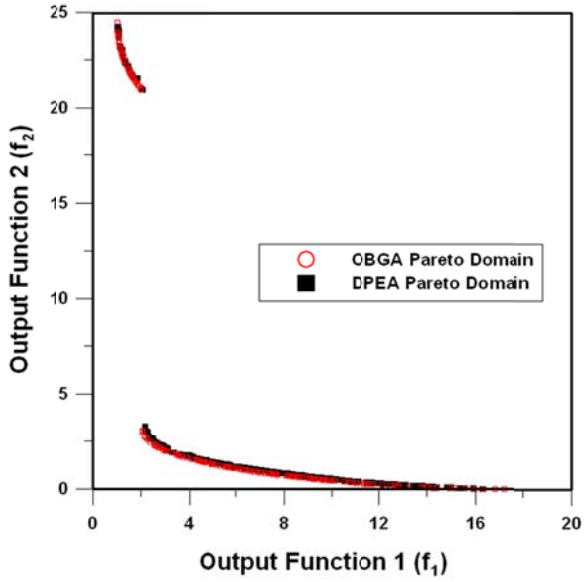

Figure 5: Output space for both the DPEA and OBGA Pareto domains for Problem 2.

In the input space, it is apparent that the OBGA Pareto domain is better defined than the DPEA Pareto domain. During the simulations performed in this study, it was found that this

difference became less pronounced at larger population sizes. The Pareto domain for NSGA-II (not shown) was better defined than the DPEA Pareto domain, but not as well defined as the OBGA Pareto domain. When the objectives are considered, this difference in accuracy is less visibly apparent both for the OBGA (shown) and NSGA-II (not shown). To gain a better understanding of the accuracy of the three algorithms, a more rigorous method is therefore necessary.

One possible method to evaluate the accuracy of the Pareto domain approximation is to calculate the distance of each point in a population from the true Pareto domain. The distance from the true Pareto domain was calculated for solutions in the DPEA, NSGA-II and OBGA Pareto domains for Problem 2. A tolerance of 10^{-4} was used for the OBGA procedure and a population size of 125 was used for the genetic algorithms. For each point in the Pareto domain for each of the three methods, the minimum distance to the true Pareto domain was calculated. The relative frequency of occurrence, defined by dividing the distribution in 10 segments, for varying values of the distance from the true Pareto domain is shown in Figure 6, for each of the three methods.

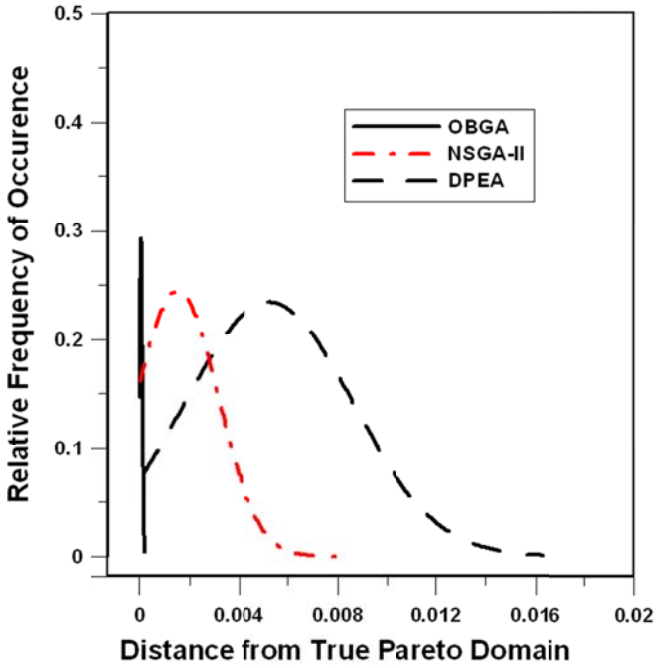

Figure 6: Relative frequency of occurrence as a function of distance from the true Pareto domain for each of the three methods.

Figure 6 clearly demonstrates that the OBGA approximated the Pareto domain with the highest accuracy for Problem 2. The average distance from the true Pareto domain for the OBGA is much smaller than that for NSGA-II and DPEA, and the distribution of distances from the true Pareto domain is also much wider for both DPEA and NSGA-II. Results also show that the accuracy of NSGA-II is higher than that of DPEA. Finally, Figure 6 confirms that the OBGA leads to a better understanding of the final accuracy for the approximated Pareto domain. The mean distance from the true Pareto domain for the OBGA was equal to 7.45×10^{-5}, confirming that the Pareto domain has been approximated within a tolerance of 10^{-4} for each of the two objectives.

Unfortunately, this method for evaluating the accuracy of the Pareto domain is not possible for many problems (such as Problems 4 in this study) as the true Pareto domain cannot be determined. Instead, the relative accuracy of each of the algorithms was evaluated in this study. This was done by combining the Pareto domains obtained from the two algorithms that were being compared. Initially the two populations consist strictly of non-dominated points. However, after the two populations are combined and the dominance test is performed on the combined population, some of the points may become dominated. If, for example, when OBGA is compared to DPEA it is found that the majority of the new dominated points originated from the DPEA population, then it can be stated that the accuracy of the DPEA is lower than that of the OBGA. This relative dominance test was therefore performed on the combined populations obtained with the different methods for Problem 2. Figure 7 presents the percentage of points that were dominated in DPEA and NSGA-II Pareto domains relative to the OBGA Pareto domain and vice versa as a function of the final tolerance value. For this comparison, a population size of 125 points was used.

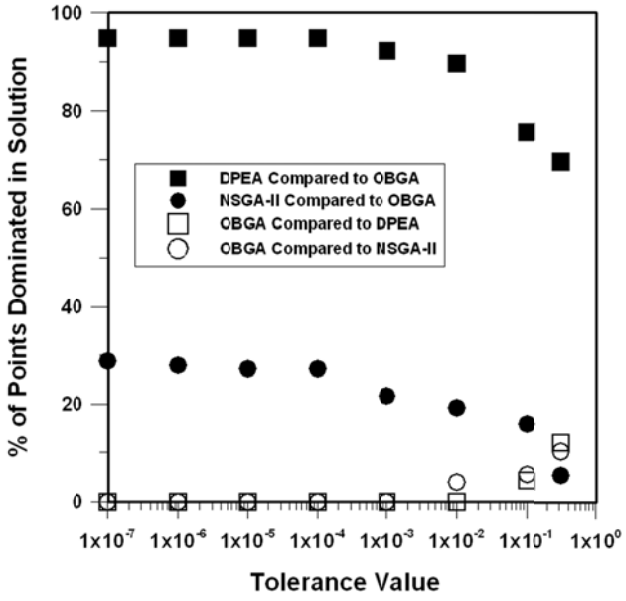

Figure 7: Percentage of points that were dominated for the DPEA and NSGA-II Pareto domains relative to the OBGA as a function of the tolerance value (closed symbols), and vice versa (open symbols).

Results show that for almost all tolerance values, the OBGA determined a more accurate Pareto domain than both DPEA and NSGA-II, as evidenced by the percentage of dominated points found in the genetic algorithms when compared to the OBGA. Results clearly show that lower tolerance values led to a more accurate Pareto domain for the OBGA, with a greater percentage of dominated points identified in the Pareto domains obtained with the genetic algorithms. Finally, it was found that NSGA-II Pareto domains were more accurate than DPEA Pareto domains, with more dominated points being present in the DPEA Pareto domains than the NSGA-II Pareto domains relative to the OBGA.

The three algorithms were also compared in terms of computation time and the number of objective function calls required in locating the Pareto domain. For this comparison (and all other comparisons in this study) the computation time and the number of objective function calls required in the initial population and the numerical derivatives for the OBGA are also included in the totals. Figure 8 presents the computation time and the number of objective function calls required to obtain the Pareto domain for Problem 2 using the OBGA. Table 2

gives the corresponding computation time and the number of objective function calls for the two genetic algorithms.

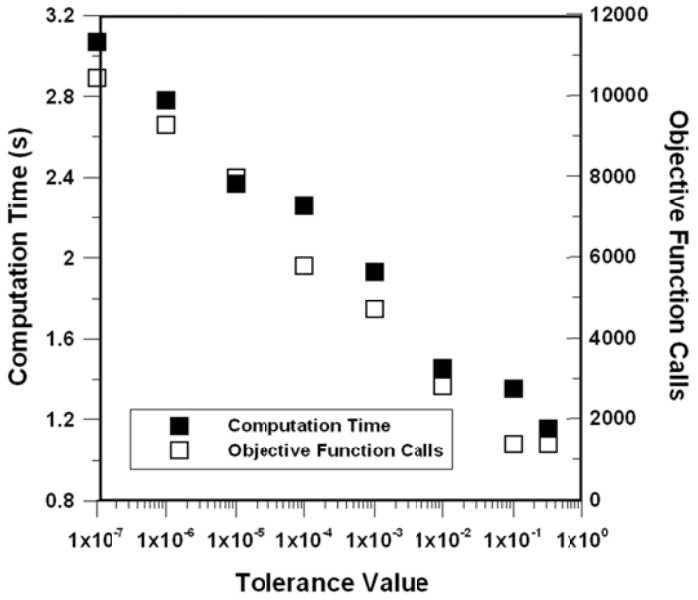

Figure 8: Computation time and the number of objective function calls for the OBGA.

Table 2: Computation time and number of objective function calls for NSGA-II and DPEA.

Method	Computation Time (s)	Objective Function Calls
NSGA-II	2.2	1207
DPEA	1.3	940

As expected, decreasing the tolerance value led to an increase in the computation time and in the number of objective function calls required. At small tolerance values, the computation time required for the OBGA was larger than both NSGA-II and DPEA. For large tolerance values, NSGA-II required the largest computation time, and DPEA and OBGA required

similar computation time. For all tolerance values, the OBGA required a greater number of objective function calls than NSGA-II and DPEA.

From the results presented in Figures 7 and 8, a preferred tolerance value can be determined for Problem 2. Figure 7 shows that at small tolerance values, the percentage of dominated points in the genetic algorithms for the combined populations remains fairly constant. When the tolerance becomes larger than 10^{-4}, the percentage of dominated points in the genetic algorithms begins to decrease significantly. In Figure 8, it was shown that larger tolerance values led to a decrease in computation time and in the number of objective function calls. Based on these findings a tolerance value of 10^{-4} was identified as giving the best compromise between high accuracy for the Pareto domain, a low computation time and a low number of objective function calls. This tolerance value was also identified as giving the best compromise for each problem used in this study. It is important to note that although decreasing the tolerance value leads to a more accurate Pareto domain, the increased accuracy may not translate to a change in operating conditions in a chemical process. Because of the intrinsic variability in processes and the difficulty in accurately measuring some process variables, it may not be possible to demonstrate the high accuracy of the OBGA Pareto domain in the specified chemical process. Nevertheless, OBGA will invariably provide a better Pareto domain even for a reasonable tolerance value which in the case of Problem 2 is as large as 0.1.

5.2 Effect of the Population Size Used in the OBGA

The effect of the population size on the performance of the OBGA was also examined. Each Pareto domain obtained using the OBGA was compared to the Pareto domains obtained using NSGA-II and DPEA with a population size of 125. Problem 2 was also used for this test to compare DPEA, NSGA-II and the OBGA. For all comparisons performed, the final tolerance used in the OBGA was set to 10^{-4} as determined in Section 5.1. It should be noted that in all tests performed with this tolerance level, no points in the OBGA Pareto domains were dominated by the two genetic algorithms. Figure 9 presents the percentage of points in the DPEA and NSGA-II Pareto domains that were dominated relative to the OBGA as a function of the population size.

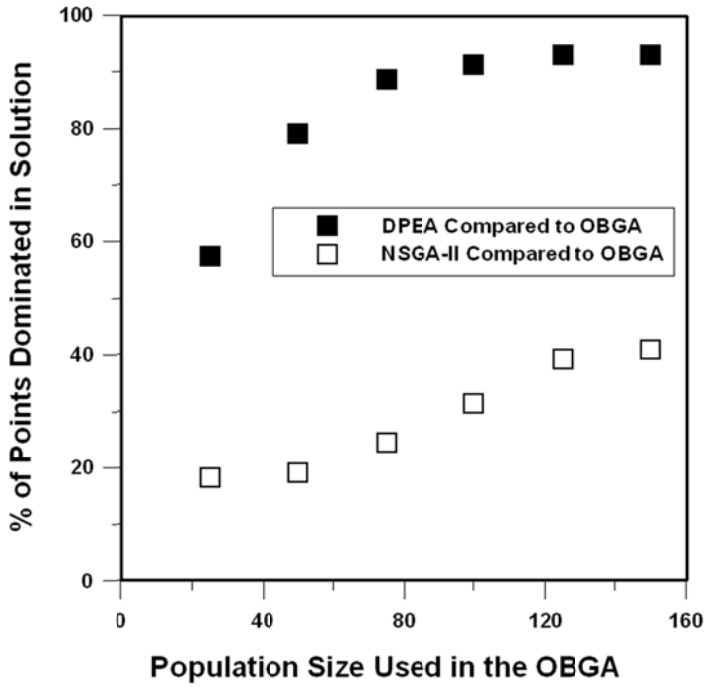

Figure 9: Percentage of points that were dominated for the DPEA and NSGA-II Pareto domains relative to the OBGA as a function of the population size for Problem 2. Population size of the genetic algorithms is 125.

Once more, the results show that the OBGA produced a more accurate Pareto domain than NSGA-II and DPEA. These results also show that larger population sizes led to a more accurate Pareto domain in the OBGA relative to the genetic algorithms. Again it was found that the NSGA-II Pareto domains were more accurate than the DPEA Pareto domains.

Figure 10 presents the computation time and the number of objective function calls required to obtain the Pareto domain using the OBGA. Again the computation time and the number of objective function calls for the two genetic algorithms corresponding to a population size of 125 are presented in Table 2. Results of Figure 10 show that increasing the population size led to an exponential increase in both the computation time and the number of objective function calls.

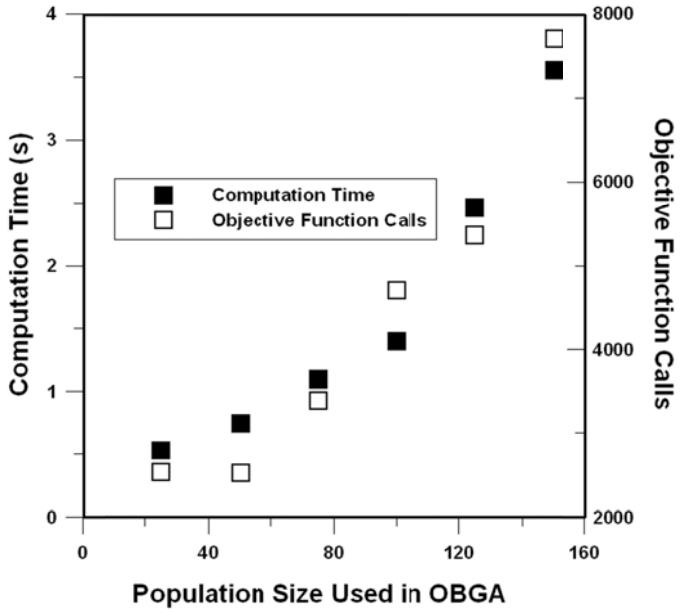

Figure 10: Computation time and objective function calls for the OBGA for Problem 2. Population size of the genetic algorithms is 125.

Figures 9 and 10 can be used to determine the preferred population size for the OBGA. Figure 9 shows that at the largest population sizes, the percentage of dominated points in the genetic algorithms remained fairly constant. When the population size used in the OBGA decreased to approximately one half of the corresponding genetic algorithm population size, the percentage of dominated points began to decrease. As depicted in Figure 10, for small population sizes, an increase in the population size led to a small increase in the computation time, and the number of objective function calls. When the population size used in the OBGA exceeded approximately one half of the corresponding genetic algorithm population size, the computation time and number of objective function calls began to increase significantly. It can also be seen from Figure 10 and Table 2 that below this population size, the computation time for the OBGA was smaller than the computation time required for NSGA-II, and comparable to DPEA. Above this population size, the computation time for the OBGA became larger than both NSGA-II and DPEA. Based on these findings, it was determined that an OBGA population size one half of the corresponding genetic algorithm population size led to the best compromise between a high accuracy for the Pareto domain, a low computation time and a low number of objective function calls.

5.3 Performance of the OBGA for the Four Problems

As a final evaluation of the OBGA performance, analysis was extended by varying the NSGA-II and DPEA population size and using an OBGA population half the size of the corresponding genetic algorithm population. A final tolerance of 10^{-4} for the OBGA was used for all comparisons. This tolerance value corresponds to a good compromise between high accuracy, low computation time and low objective function calls for all problems used in this study. For this analysis the performance of DPEA, NSGA-II and OBGA was compared for the four problems identified in Section 4. For Problems 1-3 the magnitude of the objective functions were similar, and scaling was therefore not necessary. For Problem 4 the productivity and gluconic acid production objectives were found to demonstrate a fairly large difference in magnitude, and scaling was therefore performed based on the initial population as discussed in Section 3.3. For the four problems, no points in the OBGA Pareto domains were dominated by the two genetic algorithms. Figures 11 and 12 present the percentage of points that were dominated in the DPEA and NSGA-II Pareto domains relative to the OBGA as a function of the genetic algorithm population size. The results for the four problems confirm that the OBGA led to a more accurate Pareto domain than NSGA-II and DPEA, with points being dominated in all DPEA and NSGA-II Pareto domains, and none of the OBGA Pareto domains.

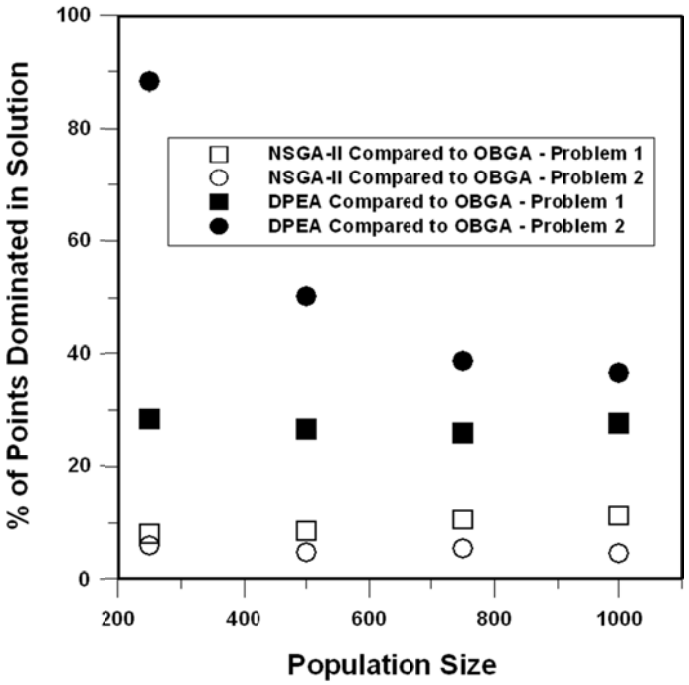

Figure 11: Percentage of points that were dominated for the DPEA and NSGA-II Pareto domains relative to the OBGA as a function of the genetic algorithm population size for Problems 1 and 2.

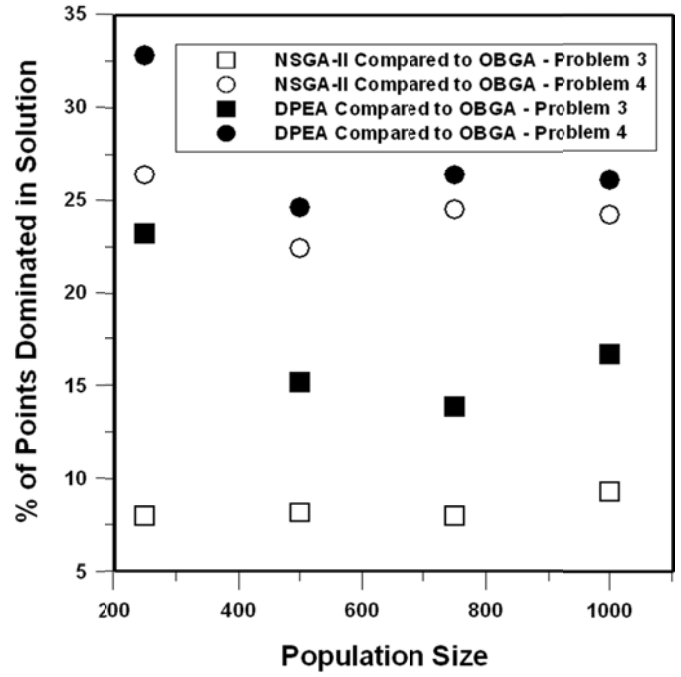

Figure 12: Percentage of points that were dominated for the DPEA and NSGA-II Pareto domains relative to the OBGA as a function of the genetic algorithm population size for Problems 3 and 4.

The effect of the population size on the accuracy of the OBGA was shown to depend on the problem and the algorithm used. For Problems 2, 3 and 4, the accuracy of the OBGA relative to DPEA is highest when the population size is low. If the selected population size used in the DPEA is too small, the obtained result will not be an accurate representation of the Pareto domain. Therefore a large percentage of the population will be dominated points relative to the more accurate OBGA Pareto domain. For Problems 3 and 4, this effect was only evident for the smallest population size of 250. For Problem 2, this effect was evident for a population size of 250, 500, and 750, as the percentage of dominated points in the DPEA solution decreased with increasing population size. When the population size was increased from 750 to 1000, the percentage of dominated points in the DPEA Pareto domain only decreased slightly, suggesting that the population size of 750 was sufficiently large for this algorithm to approximate the Pareto domain. Since this trend is not present in the NSGA-II Pareto domains, it is apparent that NSGA-II produces a more accurate Pareto domain relative to DPEA, even when the population size is small. This trend is also not present in the DPEA Pareto domains for Problem 1. This is most likely because the objective function was relatively simple, and 250 points were sufficient in producing an accurate Pareto domain.

For all comparisons performed when the genetic algorithm population sizes were sufficiently large to produce an accurate Pareto domain, the percentage of points dominated in the genetic algorithms was not found to be a function of the population size. Small variation occurred in the percentage of dominated points in the genetic algorithms as the population size was changed, as shown in Figures 11 and 12. This is most likely due to random variation present in both the genetic algorithms and the OBGA. For example, with a population size of 750 points for Problem 1, it was found that the standard deviation for the percentage of points dominated in both NSGA-II and DPEA was equal to approximately 1%. Figures 10 and 11 show that when the genetic algorithm population sizes were sufficiently large, the percentage of dominated points in the genetic algorithms varied by no more than approximately 3% with changing population size, for a given problem and genetic algorithm. It can therefore be stated that the accuracy of the Pareto domain was not a function of

population size when the population size was sufficiently large to produce an accurate Pareto domain.

Table 3 presents the computation time and the number of objective function calls required to obtain the Pareto domain for the three optimization methods and the four problems. Using a reduced population size in the OBGA led to significantly reduced computation time. For Problems 1, 2 and 3 the computation time for the OBGA was significantly less than the computation time required for the two genetic algorithms, for all population sizes, and yet the Pareto was defined more accurately. Although computation time was reduced, the number of objective function calls required to locate the Pareto domain was the greatest for the OBGA in the majority of comparisons performed. This is because the objective functions for Problems 1, 2 and 3 are relatively simple, and the increase in computation time resulting from a greater number of objective function calls is smaller than the computation time required for sorting each generation of points in the genetic algorithms. For Problem 4, which is a more complex objective function, the computation time for the OBGA was found to be greater than the genetic algorithms used for comparison.

For all four problems it can also be seen that that the efficiency of the OBGA relative to the genetic algorithms is greatest at large population sizes. Results of Problems 1, 2 and 3 show that the computation time for the OBGA is significantly lower than for NSGA-II and DPEA at large population sizes, whereas the computation time for the OBGA is comparable to DPEA at a population size of 250. Similarly for Problem 4, for large population sizes the OBGA requires only slightly more computation time than NSGA-II, whereas at small population sizes the OBGA requires significantly more computation time than both NSGA-II and DPEA. The effect of reducing the population size is therefore much more significant at a population size of 1000 points than at a population size of 250 points.

Table 3: Computation time and objective function calls for the OBGA, NSGA-II, and DPEA.

		Computation Time (S)			Objective Function Calls		
Problem	Initial Points	DPEA	NSGA-II	OBGA	DPEA	NSGA-II	OBGA
1	250	1.9	23.0	1.6	1298	5936	4511
	500	7.3	75.0	5.8	2536	13038	9032
	750	17.7	133.7	11.6	3824	18766	13009
	1000	32.1	251.8	19.6	5174	26940	17544
2	250	2.8	6.7	2.5	1658	2247	8659
	500	16.9	18.5	6.4	4758	4206	11594
	750	22.0	66.6	11.4	5305	6170	18331
	1000	53.6	63.5	22.9	8589	8070	20576
3	250	2.4	7.7	2.5	994	2152	5029
	500	9.3	37.0	5.2	2302	5029	7703
	750	22.0	66.6	11.4	3552	9598	10764
	1000	44.2	173.9	20.7	5100	13904	15385
4	250	35.7	55.9	125.1	946	1427	4077
	500	88.7	156.3	251.6	2337	3753	8246
	750	148.1	253.1	485.7	3757	5943	12155
	1000	216.8	432.6	485.7	5233	9193	15638

5.4 Limitation to Real Variables

The two genetic algorithms used in this investigation to assess the performance of the OBGA have the ability to be used for a wide variety of problems, including systems where integer decision variables and criteria are involved. On the other hand the OBGA, as is the case with most other gradient-based algorithms, cannot be used when one or more of the input variables are strictly limited to integer values. This limitation was verified by restricting one of the input variables to integer values for all four problems used in this study. It was found that limiting an input variable to integer values led to inaccuracies in the OBGA procedure. It was therefore not possible to move solutions toward the Pareto domain with the OBGA procedure. On the other hand, when one or more of the objective function values was strictly limited to integer values, the OBGA procedure was capable of moving solutions towards the Pareto domain, but with reduced accuracy relative to the case where only real variable objective functions are considered.

6.0 Conclusions

In this study, an objective-based gradient algorithm (OBGA) has been proposed. For the four problems considered in this investigation, the OBGA systematically produced a more accurate Pareto domain than the DPEA and NSGA-II MOO techniques. For relatively simple problems, the OBGA led to a decrease in computation time despite an increase in the number of objective function calls, if a reduced population size was used in the OBGA. For a more complex objective function (Problem 4) the OBGA led to an increase in computation time, despite a reduction of the OBGA population size.

From these results, it can be concluded that for an optimization problem the recommended algorithm will depend on the specified process model. If the model is relatively simple, it is always recommended to use the OBGA with a reduced population size. For complex process models, the computation time required for the OBGA may be too large if the available computation time is limited. OBGA is also not recommended for a process model with integer decision variables. Even if for a given problem the computation time could be reduced by using genetic algorithms, the OBGA still provides the best approximation of the Pareto domain, and the greatest understanding of the final accuracy obtained.

7.0 References

Agarwal A, Gupta SK. Jumping gene adaptations of NSGA-II and their use in the multi-objective optimal design of shell and tube heat exchangers. Chem. Eng. Res. Des. 2008; 86: 123-139.

Arnold D, Salomon R. Evolutionary Gradient Search Revisited. IEEE Trans. On Evol. Comput. 2007; 11: 480-95.

Bosman P, de Jong E. Exploiting Gradient Information in Numerical Multi-Objective Evolutionary Optimization. In: Beyer H, editor. Genetic And Evolutionary Computation Conference 2005: Proceedings of the 2005 conference on Genetic and Evolutionary Computation; 2005 June 25-29; Washington DC, USA. USA: Association for Computing Machinery, Inc; 2005. p. 755-62.

Brown M, Smith RE. Directed Multi-Objective Optimisation. IJCSS 2005; 6: 3-17.

Deb K. Multi-objective optimization using Evolutionary Algorithms. England: John Wiley & Sons, Ltd; 2001.

Deb K, Pratap A, Agarwal S, Meyarivan T. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. IEEE Trans. On Evol. Comput. 2002; 6: 182-97.

Deb K, Agarwal RB. Simulated Binary Crossover for Continuous Search Space. Complex Syst. 1995; 9: 114-48.

Fonteix C, Massebeuf S, Pla F, Kiss L. Multicriteria optimization of an emulsion polymerization process. Eur. J. Oper. Res. 2004; 153: 350–9.

Ghose T, Ghosh P. Kinetic analysis of gluconic acid production by *Pseudomonas ovalis*. J. Appl. Chem. Biotech 1976; 26: 268-77.

Halsall-Whitney H, Thibault J. Multi-objective optimization for chemical processes and controller design: Approximating and classifying the Pareto domain. Comput. Chem. Eng 2006; 30: 1155-68.

Lalonde L., Multiobjective Optimization Algorithm Benchmarking and Design under Parameter Uncertainty, August 2009, M.A. Sc. Thesis, Queen's University.

Lam C, Dang C, Au K. A GA/gradient hybrid approach for injection moulding condition optimisation. Eng. Comput. 2006; 21: 193-202.

Li C, Zhang X, Zhang S, Suzuki K. Environmentally conscious design of chemical processes and products: Multi-optimization method. Chem. Eng. Res. Des. 2009; 87: 233-243.

Logist F, Van Erdeghem PMM, Van Impe JF. Efficient deterministic multiple objective optimal control of (bio)chemical processes. Chem. Eng. Sci. 2009; 64: 2527-2538.

Mokeddem D, Khellaf A. Optimal Solutions of Multiproduct Batch Chemical Process Using Multiobjective Genetic Algorithm with Expert Decision System. J. Autom. Methods Manage. Chem. 2009; 2009: 1-9.

Poloni C, Giurgevich A, Onesti L, Pediroda V. Hybridization of a multi-objective genetic algorithm, a neural network, and a classical optimizer for a complex design problem in fluid dynamics. Comput. Method in Appl. M. 2000; 186: 403-20.

Salomon R. Evolutionary Algorithms and Gradient Search: Similarities and Differences. IEEE Trans. On Evol. Comput. 1998; 2: 60; 45-55.

Tarafder A, Rangaiah GP, Ray AK. Multiobjective optimization of an industrial styrene monomer manufacturing process. Chem. Eng. Sci. 2005; 60: 347-363.

Thibault J. Net Flow and Rough Sets: Two Methods for Ranking the Pareto Domain. In: Rangaiah G, Editor. Advances in Process Systems Engineering – Vol. 1: Multi-Objective Optimization: Techniques and Applications in Chemical Engineering. Singapore: World Scientific Publishing; 2009. p. 189-236.

Viennet R, Fonteix C, Marc I. Multicriteria optimization using a genetic algorithm for determining a Pareto set. Int. J. Syst. Sci. 1996; 27: 255-60.

Chapter 2

Multi-Objective Optimization of an Ethylene Oxide Reactor

Allan Vandervoort, Jules Thibault and Yash Gupta

Abstract

In this study, multi-objective optimization is performed for a reactor producing ethylene oxide from ethylene. The optimization considered three objectives: the maximization of the ethylene oxide production and selectivity, and the maximization of a safety factor related to the presence of oxygen in the reactor. The Pareto domain for this optimization problem was first approximated using the Objective-Based Gradient Algorithm, and the Pareto-optimal solutions were ranked using the Net-Flow procedure to determine the best operating conditions. From the optimization results, it is recommended that the ethylene oxide reactor be operated at high inlet pressure and gas temperature, and low inlet volumetric gas flowrate and chemical reaction moderator concentration. These operating conditions led to the highest ranked compromise solution, balancing the trade-off between each of the three objectives. Finally, it was found that variation in the inlet pressure and volumetric gas flowrate could readily lead to operating conditions outside of the Pareto domain, and these input variables should therefore be carefully controlled throughout operation of the reactor.

Journal Publication: International Journal of Chemical Reaction Engineering
Publisher: The Berkeley Electronic Press
Publication Status: Submitted

1.0 Introduction

Ethylene oxide is an important chemical, used industrially as an intermediate in the synthesis of glycol (Lahiri et al., 2009), antifreeze and polyester fibers (Aryana et al., 2009). Industrial ethylene oxide plants produce revenues totaling several billion dollars each year (Stegelmann et al., 2004) and the modeling and optimization of these plants have been studied by multiple authors (Aryana et al., 2009; Cornelio et al., 2006; Galan et al., 2009; Lahiri et al., 2009; Stegelmann et al., 2004; Zhou et al., 2005).

The study of ethylene oxide reactors has led to significant improvement in the understanding of the reaction process. Numerous variables have been identified which affect the performance of the reactor, and multiple objectives used to evaluate the performance of the reactor have also been identified. Unfortunately, current optimization studies are limited as they do not consider these multiple objectives simultaneously, or the trade-off between each objective. For this reason multi-objective optimization (MOO) of the ethylene oxide reactor was performed in this study. MOO techniques are used to optimize multiple objectives simultaneously, and the trade-off between each objective variable can be well understood.

In order to perform MOO on the ethylene oxide reactor a process model is required. Several previously developed process models were shown to fit well, real plant data (Lahiri et al., 2009; Aryana et al., 2009; Galan et al., 2009). Only the model developed by Galan et al. (2009) provides the plant data for comparison in a tabular form. This model also has the advantage of including parameters that can be adjusted to fit the plant data, which accounts for variation in the gas and coolant property estimates, as well as the chosen kinetic model. Finally, this model considers both the desired reaction, as well as two competing side reactions. Several models developed for this reactor have only considered one side reaction (Lahiri et al., 2009; Zhou et al., 2005; Cornelio et al., 2006).

2.0 Reactor Details

In the reactor model developed by Galan et al. (2009) ethylene oxide is produced from the partial oxidation of ethylene. Two undesirable reactions produce carbon dioxide and water:

the combustion of ethylene and the oxidation of ethylene oxide. The reactor is a shell and tube vessel, where the tubes are packed with porous catalyst and the coolant, Mobiltherm 306, flows in the shell side of the vessel. Silver is the most commonly used catalyst for this reaction both industrially and in research studies. In practice, small amounts of 1,2–dichloroethane (DCE) are also added in the feed of the reactor as a chemical reaction moderator. The chemical reaction moderator is adsorbed on catalyst active sites (Galan et al., 2009), and this decreases the rate of ethylene combustion more than the rate of ethylene partial oxidation (Lahiri et al., 2009). The addition of DCE therefore increases selectivity for ethylene oxide. Oxygen is added to the reactor as air, and nitrogen is therefore present in the reaction mixture as an inert gas. The composition of nitrogen and oxygen in the inlet gas stream is not equal to the composition normally found in air, as a portion of the product stream is recycled in industrial processing plants (Turton et al., 2009). Due to the recycle stream, carbon dioxide, water, and small amounts of ethylene oxide are also present in the reactor inlet.

2.1 Kinetic Model

In the ethylene oxide reactor partial oxidation of ethylene, combustion of ethylene, and oxidation of ethylene oxide occur. These reactions are shown in Equations (1) – (3) with their respective heats of reaction (Galan et al., 2009).

$$C_2H_4 + \frac{1}{2}O_2 \rightarrow C_2H_4O \qquad \Delta H_{rxn,\,1} = \text{-105 J/mol} \qquad (1)$$

$$C_2H_4 + 3O_2 \rightarrow 2CO_2 + 2H_2O \qquad \Delta H_{rxn,\,2} = \text{-1324 J/mol} \qquad (2)$$

$$C_2H_4O + \frac{5}{2}O_2 \rightarrow 2CO_2 + 2H_2O \qquad \Delta H_{rxn,\,3} = \text{-1219 J/mol} \qquad (3)$$

The reaction rate expressions are taken from two sources. The reaction rate equations for the partial oxidation of ethylene (R_1) and the combustion of ethylene (R_2), shown in Equations (4) and (5), were determined by Petrov et al. (1986). The rate equation for the oxidation of ethylene oxide (R_3), given in Equation (6), was proposed by Eliyas et al. (1988).

$$R_1 = \frac{k_1 P_{O_2} P_{C_2H_4} - k_2 P_{O_2} P_{C_2H_4} P_{DCE}^{\ 0.19}}{1 + k_5 P_{O_2} + k_6 P_{C_2H_4}} \tag{4}$$

$$R_2 = \frac{k_3 P_{O_2} P_{C_2H_4} - k_4 P_{O_2} P_{C_2H_4} P_{DCE}^{\ 0.07}}{1 + k_5 P_{O_2} + k_6 P_{C_2H_4}} \tag{5}$$

$$R_3 = \frac{k_7 P_{O_2} P_{C_2H_4} - k_8 P_{O_2} P_{C_2H_4} P_{DCE}^{\ 0.6}}{\left(1 + k_9 P_{O_2} + k_{10} P_{C_2H_4}^{\ 0.5} + k_{11} P_{C_2H_4O} + k_{12} P_{C_2H_4} P_{O_2}^{\ -0.5}\right)^2} \tag{6}$$

Where P_j represents the partial pressure of each species in the reactor.

The rate constants, k_m ($m \in [1,12]$), are calculated using the Arrhenius equation, as shown in Equation (7). The values of the activation energy (E_A) and the pre-exponential factor (A) are shown in Table 1.

$$k_m = A\, e^{-\frac{E_A}{RT_s}} \tag{7}$$

Table 1: Kinetic parameters for the ethylene-oxide production.

Rate Constant	A	E_A(J/mol K)	Units of A and k_m
k_1	6.87 x10^0	3.87 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-2}$
k_2	1.07 x10^2	4.36 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-3}$
k_3	1.06 x10^1	3.68 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-2}$
k_4	3.96 x10^1	3.96 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-3}$
k_5	1.35 x10^{-6}	4.02 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-1}$
k_6	3.11 x10^{-8}	6.29 x10^4	mol h^{-1} g-$cat^{-1}$$atm^{-1}$

k_7	1.03 x10^{-3}	9.20 x10^{1}	mol h^{-1} g-$cat^{-1}$$atm^{-2}$
k_8	2.00 x10^{-1}	1.17 x10^{1}	mol h^{-1} g-$cat^{-1}$$atm^{-3}$
k_9	4.25 x10^{-5}	1.50 x10^{2}	mol h^{-1} g-$cat^{-1}$$atm^{-1}$
k_{10}	4.59 x10^{-4}	1.09 x10^{2}	mol h^{-1} g-$cat^{-1}$$atm^{-1}$
k_{11}	3.00 x10^{-1}	4.18 x10^{1}	mol h^{-1} g-$cat^{-1}$$atm^{-1}$
k_{12}	2.00 x10^{-1}	5.02 x10^{1}	mol h^{-1} g-$cat^{-1}$$atm^{-2}$

2.2 Reactor Model Equations

The dynamic reactor model equations were developed by Galan et al. (2009). In this study, the steady-state model equations were used in the MOO procedure. The model equations are therefore shown in the steady state form below. All gas property estimates, coolant property estimates and heat transfer correlations are shown in Appendix A. Values for the reactor parameters which were not varied in the optimization procedure are given in Appendix C.

The dimensionless length *(ξ=z/L)* is used in the calculation procedure. The concentration of each component, C_j, along the length of the reactor is calculated using Equation (8) where β_{ij} is the stoichiometric coefficient of component j ($j \in [1,7]$) in reaction i ($i \in [1,3]$). It should be noted that nitrogen is inert in the reactor and DCE only modifies the reaction rates. Once the concentration of each component is calculated for a given value of ξ, the composition of each component (y_j) is calculated using Equation (9) (Galan et al., 2009).

$$\frac{dC_j}{d\xi} = \sum_{i=1}^{N_{rxn}} \beta_{ij} \hat{r}_i \frac{L}{v_g} \tag{8}$$

$$y_j = C_j \frac{RT_g}{P} \tag{9}$$

The reaction rates were adjusted to fit the plant data given in Galan et al. (2009) based on the gas and coolant property estimates and the chosen kinetic model. The adjusted rate equation ($\hat{r}_i$) can be calculated from Equation (10), where η_i is the adjusted parameter for each reaction.

$$\hat{r}_i = R_i \, \rho_p \, \eta_i \tag{10}$$

In this study the values of η_1, η_2 and η_3 associated with the rate equations (Equations (4)-(6)) were set to 0.850, 0.132, and 1.89, respectively. These values were determined by minimizing the squares of the error between the model output and the plant data given in Galan et al. (2009). The reactor parameters used for this minimization are shown in Appendix C.

The model takes into account the change in three temperatures along the length of the reactor: the surface temperature of the catalyst (T_s), the gas temperature (T_g), and the coolant temperature (T_c) (Galan et al., 2009). The changes in the gas and coolant temperatures along the length of the reactor are calculated using Equations (11) and (12), respectively. Once T_g and T_c are calculated for a given value of ξ, T_s can be calculated using Equation (13) (Galan et al., 2009).

$$\frac{dT_g}{d\xi} = \frac{1}{\rho_g \, C_{p,g}} \left[h_s \, a_s \left(T_s - T_g \right) - \frac{4U}{d_i} \left(T_g - T_c \right) \right] \frac{L}{v_g} \tag{11}$$

$$\frac{dT_c}{d\xi} = \frac{4U}{\rho_g C_{p,c}} \, \frac{N_{tubes} \, d_o}{\left(d_s^2 - N_{tubes} \, d_o^2 \right)} \left(T_g - T_c \right) \frac{L}{v_c} \tag{12}$$

$$\sum_{i=1}^{R} \hat{r}_i \left(-\Delta H_{rxn,i} \right) = h_s \, a_s \left(T_s - T_g \right) \tag{13}$$

The model also considers the pressure drop along the length of the reactor, which is calculated using Equation (14).

$$\frac{dP}{d\xi} = -f\,L\frac{\rho_g\, v_g^2}{d_p} \tag{14}$$

In this study, Equations (11)-(14) were integrated using Euler's method, with a dimensionless length step size of 0.001.

3.0 Optimization Techniques

3.1 MOO Discussion

In MOO, multiple objectives and multiple input variables are considered in determining the optimum operating conditions. MOO techniques primarily consist of two main steps: locating all of the possible candidates for the optimum solution, i.e. circumscribing the Pareto domain, and then choosing the best of the candidate solutions.

3.2 Approximating the Pareto Domain

3.2.1 Pareto Domain

The Pareto domain consists of all non-dominated solutions within the feasible solution domain. For a general definition of dominance two points are considered, P_1 and P_2. Each point is comprised of n input variables ($x_1, x_2, x_2 \ldots x_n$) and m output (objective or performance criteria) values ($f_1, f_2, f_3 \ldots f_m$). For a point P_1 to dominate a point P_2, the following two conditions must hold (Deb, 2001):

- None of the criteria values, f_1 to f_m, for P_1 are worse than the corresponding criteria values, f_1 to f_m, for P_2. For example, if all output criteria values are to be maximized then no criteria values in P_1 can be smaller than the corresponding criteria value for P_2.
- At least one objective criterion for P_1 must be better than the corresponding objective criterion for P_2.

If P_1 dominates P_2 then P_2 is a dominated point and if P_1 does not dominate P_2 and P_2 does not dominate P_1, then both points are non-dominated points with respect to each other.

For a given optimization problem, only points within the Pareto domain are considered when choosing ideal operating conditions, since all points outside of the Pareto domain are dominated points and are therefore worse for all criteria than a particular point in the Pareto domain. The Pareto domain also allows the decision maker to understand the trade-off between each objective in an optimization problem.

3.2.2 The Objective-Based Gradient Algorithm

In this study, the Objective-Based Gradient Algorithm (OBGA) developed by Vandervoort (2010) was used to approximate the Pareto domain before the ideal operating conditions were chosen. The OBGA is an algorithm that progressively moves a set of points towards the Pareto domain. Unlike many methods that are used to approximate the Pareto domain, calculations are based directly on the objective space. It was shown by Vandervoort (2010) that the OBGA approximated the Pareto domain with higher precision than the Non-Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) which is commonly used in MOO studies. The higher precision of the OBGA is ideal for locating the best set of candidate solutions.

For the OBGA, the calculation begins from a specific point in an initial population. The approximate direction that is desired to move objective functions towards the Pareto domain ($\underline{df}$) is known for an optimization problem, and therefore can first be specified. The required changes in the input values ($\underline{dx}$) are then calculated using Equation (15) if the number of inputs is equal to the number of outputs or Equation (16) if the number of inputs is not equal to the number of outputs (Equatoon (16) is the pseudo inverse). It should be noted that if the number of inputs are equal to the number of outputs Equations (15) and (16) will yield the same result, but Equation (15) will involve reduced computational time. New input values and the corresponding output function values are then calculated. If the chosen values of $\underline{df}$ correspond to a feasible point which does not exceed the Pareto domain, the new point obtained will dominate the original point. This calculation procedure is repeated each time a new point is obtained, until the resulting point reaches a location that is approaching the Pareto domain. Beyond this point, the next calculation will generate a dominated point. This

indicates that the Pareto domain has been reached within the given tolerance chosen for the objective functions ($\underline{df}$). In practice, to approximate the Pareto domain within the desired precision, once a dominated point is found the values of $\underline{df}$ are decreased and the process is continued until predetermined minimum values of $\underline{df}$ have been reached.

$$\underline{dx} = \underline{\underline{D}}^{-1} \, \underline{df} \tag{15}$$

$$\underline{dx} = \left(\underline{\underline{D}}^{T} \, \underline{\underline{D}}\right)^{-1} \, \underline{\underline{D}}^{T} \underline{df} \tag{16}$$

where each element in matrix $\underline{\underline{D}}$ (the Jacobian Matrix) is $\frac{\partial f_i}{\partial x_j}$ and represents the partial derivatives of the objective function i relative to input variable j, at the point of interest. Element df_i of vector $\underline{df}$ represents the desired change in the output function i and element dx_j of vector $\underline{dx}$ represents the corresponding change in the input variable j.

In order to approximate the entire Pareto domain, the OBGA is performed by moving several points from an initial population towards the Pareto domain. The initial population can be a random set of points or a partially or completely converged Pareto domain obtained using a more conventional method. In this study, a simple genetic algorithm, the Dual Population Evolutionary Algorithm (DPEA) (Halsall-Whitney et al., 2006), was used to provide the initial set of points necessary to begin the OBGA. The DPEA is a genetic algorithm which converges rapidly to a number of non-dominated solutions that are relatively close to the Pareto domain, and therefore represents an ideal initial population for the OBGA.

3.3 Optimum Solution Selection

The MOO technique Net Flow was used in this study to rank the solutions in the Pareto domain and select the best solution. The Net Flow method is a combination of two previous techniques used to rank the Pareto domain, known as ELECTRE and PROMETHEE (Thibault, 2009). In the Net Flow method each point in the Pareto domain is ranked based on

knowledge about the process provided by the decision maker. The information that must be provided by the decision marker is described below (Thibault, 2009).

1. The first parameter is the relative weight (W_k) of each objective in the optimization problem. In Net Flow, the relative weights are normalized.
2. The second parameter is known as the indifference threshold (Q_k). The indifference threshold represents the ranges over which two solutions are indiscernible to the decision maker with respect to each objective function.
3. The third parameter is known as the preference threshold (P_k). This parameter represents the difference between the objective criteria of two solutions that would cause the decision-maker to favor the better criterion.
4. The fourth parameter is known as the veto threshold (V_k). This parameter represents the difference between the objective criteria of two solutions that would cause the decision-maker to discard the entire solution with the worse criterion. If the veto threshold is exceeded the solution will be discarded, even if the solution is favorable for all other objective criteria.

In Net Flow, the four parameters are chosen so that the following property holds.

$0 \leq Q_k \leq P_k \leq V_k$

Once the parameters have been chosen for each objective function, they are used to calculate a ranking score for each point located in the Pareto domain. The procedure for calculating a ranking score for each point is shown below (Thibault, 2009).

1. First the difference between the values of each objective function, f_k, is calculated for each combination of solutions. For two given solutions, solution i and solution j, the difference is calculated using Equation (17).

$$\Delta_k[i,j] = f_k(i) - f_k(j) \quad \begin{cases} i \in [1, M] \\ j \in [1, M] \\ k \in [1, n] \end{cases} \tag{17}$$

where M represents the number of solutions in the Pareto domain, and n represents the number of objective functions in the optimization problem. For the minimization of an objective function, the value of Δ_k is used for the calculation procedure whereas $-\Delta_k$ is used for the maximization of an objective function.

2. Next the parameter known as the individual concordance index ($c_k[i, j]$) is calculated for each pair of solutions, and each objective function, using Equation (18).

$$c_k[i,j] = \begin{cases} 1 & if\ \Delta_k[i,j] \le Q_k \\ \dfrac{P_k - \Delta_k[i,j]}{P_k - Q_k} & if\ Q_k < \Delta_k[i,j] \le P_k \\ 0 & if\ \Delta_k[i,j] > P_k \end{cases} \tag{18}$$

If the difference between solutions $f_k(i)$ and $f_k(j)$ for a given objective function is less than the indifference threshold it cannot be said that either solution is preferable, and the concordance index is set to 1. If the difference is greater than the preference threshold, solution f_k(i) is preferred over f_k(j) and the concordance index is set to 0. Between these two thresholds the concordance index varies linearly between 0 and 1.

3. The weighted sum of the individual concordance indices for each objective function is then calculated. The weighted sum is referred to as the global concordance index comparing solution i to solution j ($C[i, j]$) and is calculated using Equation (19).

$$C[i,j] = \sum_{k=1}^{n} W_k c_k[i,j] \quad \begin{cases} i \in [1, M] \\ j \in [1, M] \end{cases} \tag{19}$$

4. Next the parameter known as the discordance index ($D_k[i, j]$) is calculated for each pair of solutions, and each objective function using Equation (20).

$$D_k[i,j] = \begin{cases} 0 & if\ \Delta_k[i,j] \le P_k \\ \dfrac{\Delta_k[i,j] - P_k}{V_k - P_k} & if\ P_k < \Delta_k[i,j] \le V_k \\ 1 & if\ \Delta_k[i,j] > V_k \end{cases} \tag{20}$$

If the difference between solution $f_k(i)$ and $f_k(j)$ for a given objective function is greater than the veto threshold it can be said that solution $f_k(j)$ is significantly worse than solution $f_k(i)$ and the discordance index is set to 1. If the difference is less than the preference threshold it cannot be stated that either solution is significantly better and the discordance index is set to 0. Between these two thresholds the discordance index varies linearly between 0 and 1.

5. The relative performance of each pair of solutions is then evaluated by calculating each element in the outranking matrix ($\sigma[i, j]$) using Equation (21).

$$\sigma[i,j] = C[i,j]\left(\prod_{k=1}^{n}\left[1-\left(D_k[i,j]\right)^3\right]\right) \quad \begin{cases} i \in [1, M] \\ j \in [1, M] \end{cases} \tag{21}$$

Each value of $\sigma[i, j]$ represents the quality of solution i relative to solution j for all objective functions. A value of $\sigma[i, j]$ close to 0 suggests that solution j outperforms solution i, and a value close to 1 could suggest that solution i outperforms solution j, or that solution i and solution j have similar performance.

6. A final ranking score is then calculated for each solution in the Pareto domain using Equation (22).

$$\sigma_i = \sum_{j=1}^{M}\sigma[i,j] - \sum_{j=1}^{M}\sigma[j,i] \tag{22}$$

The first term evaluates the performance of solution i relative to all other solutions, and the second term evaluates the performance of all other solutions relative to solution i. The point with the largest ranking score can be chosen as the final operating conditions, or the Pareto domain can be divided into zones, showing the regions of the best, moderate, and worst operating conditions (Thibault, 2009).

3.4 Optimization Problem

The ethylene oxide reactor model contains several possible objective functions, and many input parameters (decision variables). In previous optimization studies, each objective was optimized individually, or several objectives were combined into a weighted function which

leads to only one solution, and does not highlight the trade-offs between each objective. In these studies, the two most common objectives were the selectivity in the reactor and the rate of ethylene oxide production (Aryana et al., 2009; Galan et al., 2009; Lahiri et al., 2009; Zhou et al., 2005). In Galan et al. (2009), an explosion hazard due to the presence of oxygen in the reactor is also described. Specifically, if the oxygen concentration in the reactor is greater than 7 mol %, the explosion risk is considered very high. High oxygen consumption in the reactor is therefore desired. To account for this hazard the difference between the oxygen concentration at the reactor exit and the explosion limit of 7 mol % oxygen was added as a third objective function. All three objective functions are to be maximized.

For this optimization study, four input or decision variables were chosen which have been identified as having a significant influence on the objective functions, and being nearly non-correlated (Lahiri et al., 2009). The inlet reactor pressure (P_0) was chosen as an input variable due to its effect on the reaction rates and the gas properties in the reactor. The second input variable chosen was the amount of chemical reaction moderator present in the inlet reaction mixture ($y_{DCE,\,0}$) which affects the selectivity in the reactor. The third input variable chosen was the gas flow rate (Q_g), which affects the concentration changes along the reactor length and the heat transfer in the reactor. The final input variable chosen was the inlet gas temperature (which is also equal to the catalyst temperature at the entrance of the reactor). The inlet gas temperature ($T_{g,\,0}$) influences the reaction rates, and the gas properties in the reactor. A schematic diagram of the optimization problem is shown in Figure 1.

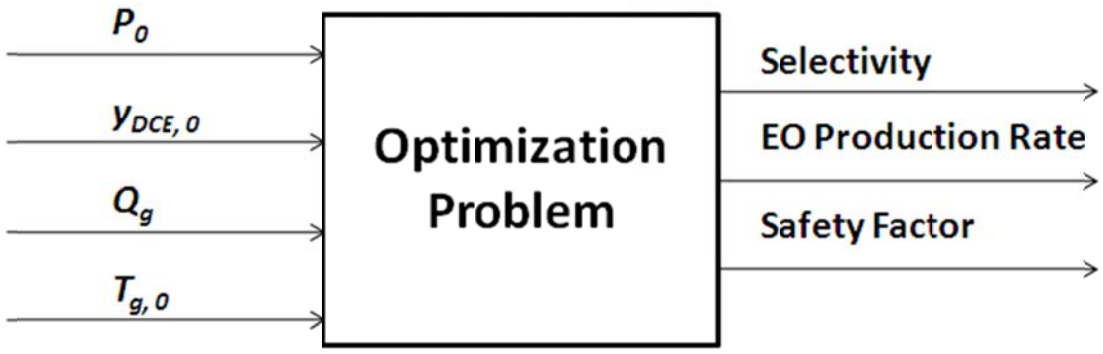

Figure 1: Optimization problem showing input variables and objective functions

Based on the operating conditions identified by Galan et al. (2009) and on the conditions specified in other ethylene oxide reactor studies (Aryana et al., 2009; Lahiri et al., 2009) values were chosen for the feasible region of each input variable. These ranges are as follow:

$P_0 \in [2, 4]$ MPa, $y_{DCE,\,0} \in [1, 1000]$ ppb, $Q_g \in [25, 55]$ $m^3\,h^{-1}$ and $T_{g,\,0} = T_{c,\,0} \in [150, 250]$ ^oC.

4.0 Results and Discussion

In the first step of this optimization process, the Pareto domain was approximated with 1000 points using the OBGA procedure to ensure a well-defined Pareto domain. Next, to determine the ranking score of each point in the Pareto domain, the preferences of a decision maker were required. For the ethylene oxide reactor optimization the highest relative weight was assigned to the safety factor to ensure the plant is operating without an explosion hazard. Ethylene oxide production was assigned the second highest relative weight, and selectivity was assigned the lowest relative weight. Threshold values were then chosen for each objective. The chosen relative weights and threshold values are shown in Table 2.

Table 2: Net Flow parameters used in optimization study.

	Net Flow Parameter			
Objective Function	W_k	Q_k	P_k	V_k
EO Production Rate	0.35	1 tonne h^{-1}	2 tonnes h^{-1}	3 tonnes h^{-1}
Selectivity	0.25	0.50 %	1 %	2 %
Safety Factor	0.40	0.25 %	0.5%	1 %

The Net-Flow calculation procedure was then performed for each solution in the Pareto domain, and the solutions were sorted based on the ranking score. The first 5 % of points

were placed into the highest ranking zone, the next 45 % of points were placed in the middle ranking zone, and the final 50 % were placed in the lowest ranking zone. The ranked Pareto domain is shown in Figures 2 and 3, with the ranking zones and the solution with the highest ranking score clearly indicated on both figures.

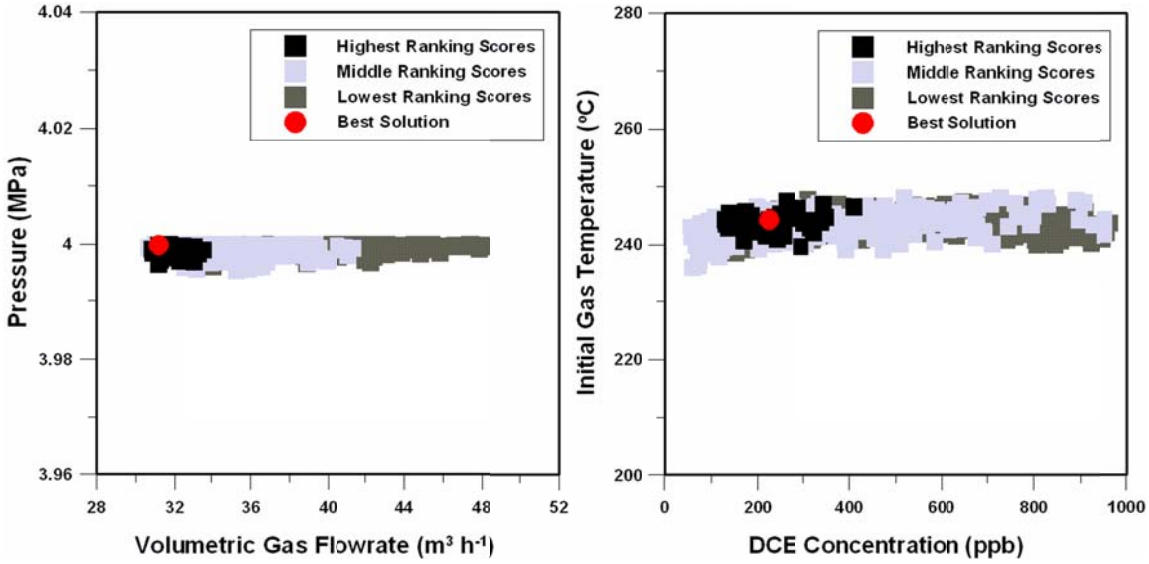

Figure 2: Input space for the ranked ethylene oxide reactor Pareto domain.

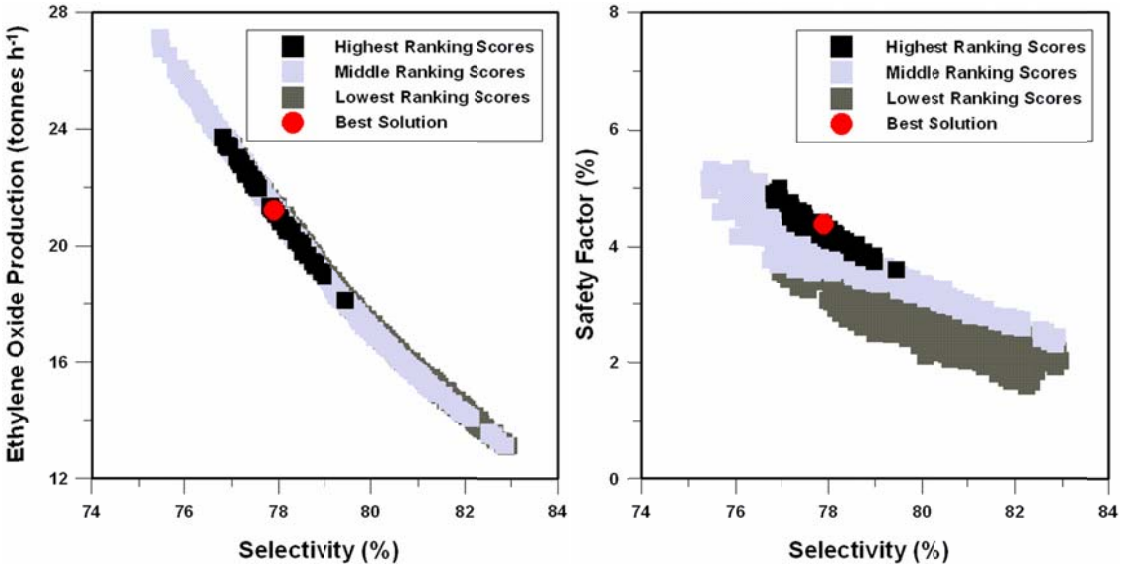

Figure 3: Output space for the ranked ethylene oxide reactor Pareto domain.

Some conclusions can be drawn based on the optimization results even before considering the ranking scores. For all points within the Pareto domain it is apparent that the inlet reactor pressure is found at its upper range. The range in inlet pressure for the optimization was 2-4 MPa, and no value less than 3.99 MPa was found in the Pareto domain. Therefore it is recommended to operate the ethylene oxide reactor at an inlet pressure as close to 4 MPa as possible, within the pressure limitations for a given reactor. The Pareto domain includes values of the volumetric gas flowrate throughout the majority of the feasible region specified by the optimization problem, with only a small range of the largest and smallest flowrate values not included in the Pareto domain. For the inlet DCE concentration the entire feasible region was included in the Pareto domain. Finally for the inlet gas temperature, only values close to the upper range of the feasible region were found in the Pareto domain, with no temperature less than 235^{o}C. These optimum ranges for the input variables are applicable to the operation of the ethylene oxide reactor, regardless of the Net-Flow parameters chosen.

Although previous ethylene oxide reactor studies have not located a Pareto domain for this optimization problem, results from several studies can be used to partially verify the Pareto domain obtained. In Figure 3, the selectivity was found to be in the range of approximately 75 % to 83 %. These values of selectivity are consistent with selectivity values reported in previous ethylene oxide reactor studies (Aryana et al., 2009; Galan et al., 2009, Lahiri et al., 2009; Zhou et al., 2005). Although ethylene oxide production values were not given in Galan et al. (2009) the Pareto domain approximated in this study includes values similar to plant data reported by Aryana et al. (2009) and Stegelmann et al. (2004). Ethylene oxide production rates determined from this optimization were higher than those reported by Aryana et al. (2009), but lower than those reported by Stegelmann et al. (2004). Both optimization studies were based on operating conditions which differed from those identified by Galan et al. (2009). For the oxygen safety factor, it was found that all optimum values contained at least 1.5 percent less oxygen than the upper safety limit.

Further recommendations for the operation of the ethylene oxide reactor can be given based on the chosen Net-Flow parameters. Figure 2 suggests that for this set of Net-Flow parameters, the reactor should be operated at the lowest values of volumetric gas flowrate

found in the Pareto domain, and relatively low values of inlet DCE concentration, to ensure high ranking scores. Values of the inlet gas temperature throughout the entire range found in the Pareto domain are present in the highest ranked solutions.

Figure 3 confirms that this optimization realized the preferences of the decision maker. The highest weight was assigned to the safety factor and production rate objectives. The highest ranked solutions include moderate to large values of safety factor and ethylene oxide production, but low to moderate values of the selectivity. The effect of the chosen Net-Flow parameters is also apparent when considering the lowest ranked solutions. The lowest ranked solutions include small values of the safety factor, small to moderate values of ethylene oxide production and large values of selectivity; again confirming that the preferences of the decision maker were realized.

The point with the highest overall ranking score is shown in Figures 2 and 3, as well as Table 3. This point includes a relatively large value of both the safety factor and the ethylene oxide production rate, but neither value is at the maximum found in the Pareto domain. This point also includes a moderate value of selectivity. The objective function values found in the highest ranked solution clearly demonstrate the trade-off which occurs when multiple objectives conflict in an optimization problem. No objective is at the maximum value found in the Pareto domain, despite the high ranking score of the solution.

Recommendations can also be made for the control scheme of the ethylene oxide reactor by considering the highest-ranked solution. In Figure 2 it is apparent that the highest ranked solution includes values of the inlet pressure and volumetric gas flowrate which are close to the boundary of the Pareto domain. It is therefore possible that small variation in the gas flowrate or a decrease in the inlet pressure could lead to operating conditions outside of the Pareto domain, which corresponds to significantly reduced reactor performance. The highest ranked solution includes values of the inlet gas temperature and DCE concentration which are not found close to the boundary of the Pareto domain. Therefore, small variations in these two input variables may lead to a small decrease in ranking score, but will not readily lead to operation outside of the Pareto domain. From these observations it is recommended that both

the inlet pressure and volumetric gas flowrate should be controlled carefully to decrease the risk of operating outside the Pareto domain.

Table 3: Optimum solution as determined by Net Flow.

Volumetric Gas Flowrate	Inlet Pressure	Inlet DCE Concentration	Inlet Gas Temperature
31.17 m^3 h^{-1}	4.000 MPa	223.1 ppb	244.2^oC
EO Production Rate	Selectivity	Safety Factor	
21.21 tonnes h^{-1}	77.89 %	4.379 %	

5.0 Conclusions

In this study, the multi-objective optimization of a reactor producing ethylene oxide from ethylene was performed. Previous optimization studies for this reactor have most often been limited to a single objective, most commonly maximizing selectivity or ethylene oxide production. In this study both of these objectives were maximized simultaneously as well as a third objective, the safety factor related to the oxygen concentration in the reactor. Four input or decision variables were used in the optimization: the inlet gas temperature, pressure, volumetric gas flowrate and chemical reaction moderator concentration. The Pareto domain was first approximated using the Objective-Based Gradient Algorithm, and each solution was assigned a ranking score using the Net-Flow procedure.

From the optimization results ideal operating conditions for the ethylene oxide reactor were identified. It is recommended that the ethylene-oxide reactor be operated at high values of the inlet pressure and gas temperature, and low values of the inlet volumetric gas flowrate and chemical reaction moderator concentration. These operating conditions lead to the best compromise or trade-off solution between each of the three objectives. In the Pareto domain, the highest ranked solution included values of both the inlet pressure and volumetric gas flowrate close to the boundary of the Pareto domain. Variation in the gas flowrate, or a decrease in the inlet pressure could therefore lead to deteriorated performance, with

operating conditions outside of the Pareto domain. It is therefore recommended to carefully control these two input variables to ensure optimum performance is maintained throughout the reactor's operation.

6.0 Appendices

Appendix A: Gas, Coolant, and Heat Transfer Equations

When estimating the properties of the reactor gas stream it was assumed that the mixture behaved as an ideal gas. This assumption allows the mass flow rate, the mass flux, and the velocity of the gas to be calculated from Equations (23), (24), and (25) respectively (Galan et al., 2009).

$$F_g = \left[\frac{P}{R T_g} MW_g \right] Q_g \tag{23}$$

$$G_g = \frac{4 F_g}{\pi d_i^2 \varepsilon N_{tubes}} \tag{24}$$

$$v_g = \frac{G_g}{\rho_g} \tag{25}$$

The coolant mass flux and velocity are found using Equations (26) and (27), where the mass flow of coolant is constant in the reactor (Galan et al., 2009). Coolant properties were taken from the industrial coolant supplier Duratherm (2010).

$$G_c = \frac{4 F_C}{\pi \left(d_s^2 - N_{tubes} d_o^2 \right)} \tag{26}$$

$$v_c = \frac{G_c}{\rho_c} \tag{27}$$

The friction factor of the gas within the reactor is used to calculate the pressure drop along the length of the reactor. The friction factor (*f*) is calculated using Equation (28), the

Reynolds number of the gas stream is calculated with Equation (29), and the characteristic diameter of the catalyst is calculated from Equation (30) (Galan et al., 2009). A value of 0.0039 m was used for the catalyst particle diameter (Cornelio et al., 2006).

$$f = \frac{1-\varepsilon}{\varepsilon^3}\left[1.75 + 150\frac{1-\varepsilon}{N_{\mathrm{Re},p}}\right] \tag{28}$$

$$N_{\mathrm{Re},p} = \frac{G_g\,\hat{d}_p}{\mu_g} \tag{29}$$

$$\hat{d}_p = 1.5\,d_p \tag{30}$$

The viscosity (in Pa s) of each component in the gas mixture was estimated from Equation (31). Similarly the thermal conductivity (in J m^{-1} s^{-1} K^{-1}) of each component in the gas mixture can be calculated from Equation (32). The coefficient in both Equations (31) and (32) were modified to provide values in S.I units. The collision integral is calculated from Equation (33) (Reid et al., 1977) and the required values of values of $\varepsilon_{k,j}$ were taken from Reid et al. (1977).

$$\mu_j = 26.69\,x10^{-7}\frac{\left(MW_{g,j}\,T_g\right)^{0.5}}{\sigma_j^{\,2}\,\phi_j} \tag{31}$$

$$\lambda_j = 0.08322\frac{\left(T_g / MW_{g,j}\right)^{0.5}}{\sigma_j^{\,2}\,\phi_j} \tag{32}$$

$$\phi_j = \frac{1.16145}{T_g^{\,0.14874\frac{T_g}{(\varepsilon/k)_j}}} + \frac{0.52487}{e^{0.7732\frac{T_g}{(\varepsilon/k)_j}}} + \frac{2.16178}{e^{2.43787\frac{T_g}{(\varepsilon/k)_j}}} \tag{33}$$

The heat capacity of the gas is calculated using Equation (34). The coefficients used in the heat capacity calculation are found in Reid et al. (1977). This equation was modified to provide heat capacity in S.I. units.

$$C_{pg,j} = 4.184 \left(\alpha_{0,j} + \alpha_{1,j} T_g + \alpha_{2,j} T_g^{\,2} + \alpha_{3,j} T_g^{\,3} \right) \tag{34}$$

The correlations used to calculate the heat transfer coefficients for various locations in the reactor are given in Equations (35)-(38) (Galan et al., 2009). The Prandtl number and the Reynolds number of the coolant can be calculated using Equations (39) and (40), respectively (Galan et al., 2009). It was assumed that the reactor walls were made of stainless steel, and thermal properties were taken from Welty et al. (2001).

$$h_s = \left(\frac{C_{p,g}\,\mu_g}{\lambda_g} \right)^{-0.67} \left[\frac{2.867}{N_{\mathrm{Re},p}} + \frac{0.3023}{N_{\mathrm{Re},p}^{\,0.35}} \right] \rho_g\, C_{p,g}\, v_g \tag{35}$$

$$h_i = 7.676 + 0.0279 \frac{\lambda_g}{\hat{d}_p} N_{\mathrm{Re},p} \tag{36}$$

$$h_o = 0.023\, N_{\mathrm{Re},c}^{\,-0.2} N_{\mathrm{Pr},c}^{\,-0.6} \rho_c\, C_{p,c}\, v_c \tag{37}$$

$$\frac{1}{U} = \frac{1}{h_i} + \frac{d_o - d_i}{2\,\lambda_w} + \frac{1}{h_o} \tag{38}$$

$$N_{\mathrm{Pr},c} = \frac{C_{p,c}\mu_c}{\lambda_c} \tag{39}$$

$$N_{\mathrm{Re},c} = \frac{G_c}{\mu_c} \left[\frac{d_s^2 - N_{tubes}\, d_o^2}{d_s} \right] \tag{40}$$

Appendix B: Variable and Subscript Definitions for the Reactor Model

Table 4: Definition of variables used in the reactor model

Variable	Definition	Unit
A	Pre-Exponential Factor	mol h^{-1} g-cat^{-1}
a_s	Gas-solid Interfacial Area Per Unit Volume of Bed	m^2 m^{-3}
C_j	Species Concentration	mol m^{-3}
C_p	Heat Capacity	J kg^{-1} K^{-1}
d_p	Catalyst Diameter	m
$\hat{d}_p$	Characteristic Diameter	m
E_A	Activation Energy	J mol^{-1} K^{-1}
F	Mass Flowrate	kg s^{-1}
f	Friction Factor	dimensionless
G	Mass Flux	kg m^{-2} s^{-1}
h	Heat Transfer Coefficient	J m^{-2} s^{-1} K^{-1}
k	Reaction Rate Constant	varies
L	Portion of Reactor Length Packed with Catalyst	m
MW	Molecular Weight	g mol^{-1}

N_{Pr}	Prandtl Number	dimensionless
N_{Re}	Reynolds Number	dimensionless
N_{tubes}	Number of Tubes in Reactor	dimensionless
N_{rxn}	Number of Reactions	dimensionless
P	Total Pressure	atm
P_j	Partial Pressure	atm
Q	Volumetric Flowrate	$m^3\ s^{-1}$
R	Ideal Gas Constant	$m^3\ atm\ K^{-1}\ mol^{-1}$
R_i	Reaction Rate	$mol\ h^{-1}\ g\text{-}cat^{-1}$
T	Temperature	K
U	Overall Heat Transfer Coefficient	$J\ m^{-2}\ s^{-1}\ K^{-1}$
v	Velocity	$m\ s^{-1}$
y	Mole Fraction	mol %
z	Distance Along Reactor	m
Greek Letters		
$\alpha_1, \alpha_2, \alpha_3$	Heat Capacity Coefficients	dimensionless
β	Stoichiometric Coefficient	dimensionless
ΔH_{rxn}	Heat of Reaction	$J\ mol^{-1}\ K^{-1}$
ε	Void Fraction	dimensionless

ε/k	Lennard-Jones Parameters	K
ξ	Dimensionless Length	dimensionless
ρ	Density	kg m^{-3}
Φ	Collision Integral	dimensionless
λ	Thermal Conductivity	J m^{-1} s^{-1} K^{-1}
η	Adjusted Reaction Rate Parameter	dimensionless
σ	Hard Sphere Diameter	Angstroms
$\hat{r}$	Adjusted Reaction Rate	mol h^{-1} m^{-3}
μ	Viscosity	kg m^{-1} s^{-1}

Table 5: Definition of subscripts used in the reactor model

Subscript	Definition
g	Gas Stream
c	Coolant
f	Final
i	Reaction Number (1-3)
j	Component Number (1-7)
m	Reaction Rate Constant Number (1-12)
p	Particle (catalyst)

s	Catalyst Surface
w	Wall
0	Inlet

Appendix C: Reactor Operating Parameters

Table 6: Reactor operating parameters

Variable	Definition	Value
d_{cat}	Catalyst Diameter	0.0039 m
d_o	Outer Tube Diameter	0.025 m
d_i	Inner Tube Diameter	0.021 m
d_s	Shell Diameter	3.3 m
F_c	Coolant Flowrate	1174 kg/s
L	Packed Reactor Length	6.5 m
N_{tubes}	# of Tubes	7750
$T_{c,\,o}$	Inlet Coolant Temperature	234^{o}C
$T_{g,\,o}$	Inlet Gas Temperature (for adjustment factor calculation)	187^{o}C
$T_{s,\,o}$	Inlet Catalyst Surface Temperature (for adjustment	187^{o}C

	factor calculation)	
$y_{CO2,\,0}$	Inlet Carbon Dioxide Composition	6.3 mol %
$y_{C2H4,\,0}$	Inlet Ethylene Composition	16.8 mol %
$y_{C2H4O,\,0}$	Inlet Ethylene Oxide Composition	10 ppb
$y_{EO,\,0}$	Inlet Ethylene Oxide Composition	0.02 mol %
$y_{H2O,\,0}$	Inlet Water Composition	0.31 mol %
$y_{O2,\,0}$	Inlet Oxygen Composition	6.7 mol %
$y_{N2,\,0}$	Inlet Nitrogen Composition	69.87 mol %
ε	Void Fraction	0.45
ρ_{cat}	Catalyst Density	1260 kg m^{-3}

7.0 References

Aryana S., Ahmadi M., Gomes V. G., Romagnoli J. and Ngian K., "Modelling and Optimisation of an Industrial Ethylene Oxide Reactor", Chemical Product and Process Modeling, 2009, 4, 1, 1-27.

Cornelio A. A., "Dynamic Modelling of An Industrial Ethylene Oxide Reactor", Indian Chemical Engineering Journal, 2006, 48, 3, 164-177.

Deb K., "Multi-objective optimization using Evolutionary Algorithms", 2001, John Wiley & Sons, Ltd, England.

Deb K., Pratap A., Agarwal S. and Meyarivan T. A., “A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II”, IEEE Transactions On Evolutionary Computation, 2002, 6, 2, 182-97.

“Compare Common Industry Fluids”, Retrieved 2010, from Duratherm: Extended Life Fluids Web site: http://www.heat-transfer-fluid.com/heat-transfer-fluid/

Eliyas A., Petrov L. and Shopov D., “Ethylene Oxide Oxidation Over a Supported Silver Catalyst. II. Kinetics of Inhibited Oxidation”, Applied Catalysis, 1988, 41, 1, 39-52.

Galan O., Gomes V. G., Romagnoli J. and Ngian K. F., “Selective Oxidation of Ethylene in an Industrial Packed-Bed Reactor: Modelling, Analysis and Optimization”, International Journal of Chemical Reaction Engineering, 2009, 7, A32, 1-26.

Halsall-Whitney H. and Thibault J., “Multi-objective optimization for chemical processes and controller design: Approximating and classifying the Pareto domain”, Computers and Chemical Engineering, 2006, 30, 6-7, 1155-68.

Lahiri S. K. and Khalfe N., “Process Modeling and Optimization of Industrial Ethylene Oxide Reactor by Integrating Support Vector Regression and Genetic Algorithm”, The Canadian Journal of Chemical Engineering, 2009, 87, 1, 118-128.

Lam C., Dang C. and Au K., “A GA/gradient hybrid approach for injection moulding condition optimisation”, 2006, Engineering with Computers, 21, 3, 193-202.

Petrov L., Eliyas A. and Shopov D., “Kinetics of Ethylene Oxidation Over a Silver Catalyst in the Presence of Dichloroethane”, Applied Catalysis, 1986, 24, 1, 145-161.

Reid R. C., Prausnitz J. M., and Sherwood T. K., “The properties of gases and liquids”, 1977, McGraw-Hill, USA.

Stegelmann C., Schiødt N. C., Campbell C. T. and Stoltze P., “Microkinetic modeling of ethylene oxidation over silver”, Journal of Catalysis, 2004, 221, 2, 630-649.

Thibault J., “Net Flow and Rough Sets: Two Methods for Ranking the Pareto Domain”, In: Rangaiah G., Editor, “Advances in Process Systems Engineering – Vol. 1: Multi-Objective Optimization: Techniques and Applications in Chemical Engineering”, 2009, World Scientific Publishing, Singapore, 189-236.

Turton R., Bailie R. C., Whiting W. B. and Shaeiwitz J. A., “Analysis, Synthesis, and Design of Chemical Processes”, 2009, Pearson Education, Inc, USA.

Welty J. R., Wicks C. E., Winson R. E. and Rorrer G., “Fundamentals of Momentum, Heat, and Mass Transfer”, 2001, Wiley & Sons, USA.

Vandervoort A., “Multi-objective optimization techniques and their application to complex chemical engineering problems”, December 2010, M.A. Sc. Thesis, University of Ottawa.

Zhou X. and Yuan W., “Optimization of the fixed-bed reactor for ethylene epoxidation”, Chemical Engineering and Processing, 2005, 44, 10, 1089-1107.

Chapter 3

A Principal Component Grid Algorithm for Approximating the Pareto Domain

Allan Vandervoort, Jules Thibault* and Yash Gupta

In this paper, principal component analysis (PCA) is incorporated into the Grid Search Approach (GSA) in order to define the Pareto domain more accurately and efficiently. The proposed Multi-Objective Optimization (MOO) algorithm is referred to as the Principal Component Grid Algorithm (PCGA) and its performance in approximating the Pareto domain is evaluated in terms of accuracy, computation time and the number of objective function calls. The performance of the PCGA is compared to the GSA and a popular MOO technique, Non-Dominated Sorting Genetic Algorithm-II (NSGA-II), using four test problems. For all problems studied, the PCGA produced a Pareto domain with greater accuracy using lower computation time and fewer objective function calls. The results from this study indicate that the PCGA is an effective and efficient algorithm for approximating the Pareto domain.

Keywords: grid search; principal component analysis; Pareto domain; multi-objective optimization

Journal Publication: Engineering Optimization
Publisher: Taylor and Francis
Publication Status: Submitted

1.0 Introduction

In complex industrial processes, several important process variables need to be optimized simultaneously to ensure that the process is operating in an efficient and cost-effective manner. This is especially important when the objectives are conflicting, such as environmental and economic concerns in a chemical process. The multiple objectives are often combined into a unique weighted objective function. This method allows finding an optimum value but does not reveal information about the tradeoff between each objective function. In addition, the weighting must be chosen carefully to ensure finding a solution that satisfies the expectation of the decision-maker. Moreover, the amalgamation of all objective functions into a unique objective function may lead to a solution that is a local optimum. For this reason further testing may be required to locate the global optimum, which may result in greater computation time (Deb, 2001).

In recent years, to circumvent the limitations associated with the weighted objective methods, multi-objective optimization (MOO) techniques have been developed which allow for several objectives to be optimized simultaneously (Deb, 2001). In MOO the best operating region, known as the Pareto domain, is identified. Several studies have proposed algorithms for approximating the Pareto domain (Deb et al., 2002; Halsall-Whitney et al., 2006; Poloni et al., 2000; Fonteix et al., 2004; Viennet et al., 1995). Of the MOO techniques used to approximate the Pareto domain, a grid-based search is one of the simplest and easiest to implement. The Grid Search Approach (GSA) also ensures a Pareto domain which evenly and fully spans the input space. However, the GSA can lead to an approximation of the Pareto domain with limited accuracy, requiring high computation time (Halsall-Whitney et al., 2006). In this study, modifications are made to the GSA using Principal Component Analysis (PCA). These modifications are performed with the goals of increasing the accuracy of the obtained Pareto domain, and reducing the required computation time.

2.0 The Pareto Domain and the Concept of Dominance

Before the decision-maker chooses a good compromise solution by considering tradeoffs between the competing criteria, the search domain can be significantly reduced by only considering solutions that would potentially be candidates in the selection of the optimal

solution. The set of retained solutions, called the Pareto domain, is obtained based on the concept of dominance.

In Figure 1, the concept of dominance and the Pareto domain are described graphically using a simple illustrative example with two output functions, f_1 and f_2, which depend on two input variables, x_1 and x_2. In this example, both functions are to be maximized. Four points are used for this illustration. Point A has the lowest values for both output functions. It can therefore be stated that point A is dominated by points B, C and D and that under no circumstances will point A be considered optimum. Although point B dominates point A, point B has lower output function values than points C and D. Therefore point B is also dominated by points C and D. When points C and D are compared to each other, both points are higher in one output function value, and lower in the other. Therefore points C and D are said to be non-dominated points. In this example, points C and D would belong to the Pareto domain if no additional points with higher values for both output functions were generated.

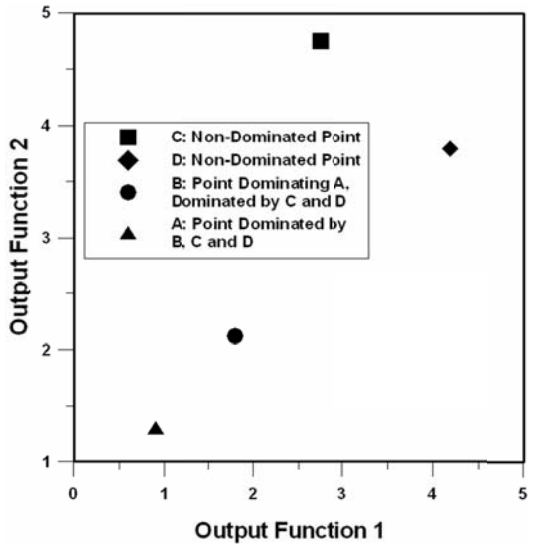

Figure 1: Illustration of the concept of dominance to define the Pareto domain.

For a general definition of dominance, consider two points, P_1 and P_2, comprised of n input variables ($x_1, x_2, x_3 \ldots x_n$) and m output (objective or performance criteria) values ($f_1, f_2, f_3 \ldots f_m$). For a point P_1 to dominate a point P_2, the following two conditions must hold (Deb, 2001):

- None of the criteria values, f_1 to f_m, for P_1 are worse than the corresponding criteria values, f_1 to f_m, for P_2. For example, if all output criteria values are to be maximized then no criteria values in P_1 can be smaller than the corresponding criteria values for P_2.
- At least one objective criterion for P_1 must be better than the corresponding objective criterion for P_2.

If P_1 dominates P_2 then P_2 is a dominated point and if P_1 does not dominate P_2 and P_2 does not dominate P_1, then both points are non-dominated points with respect to each other.

For a given optimization problem, the Pareto domain is the region within the domain of all feasible solutions that contains only non-dominated points. All points outside of the Pareto domain are dominated points and are therefore worse for all criteria than a particular point in the Pareto domain.

3.0 Current Methods for Approximating the Pareto Domain

3.1 Non-Sorting Genetic Algorithm II

The Non-Sorting Genetic Algorithm II (NSGA-II) is an MOO technique developed by Deb et al. (2002). NSGA-II is an optimization technique based on a genetic algorithm. NSGA-II is initialized by a random set of points, normally referred to as the initial population of points. The initial population and subsequent generations are progressively improved by generating new populations based on random variation and combination of points from the previous population. NSGA-II is commonly used for a variety of optimization applications (Agarwal et al., 2008; Mokeddem et al., 2009; Tarafder et al., 2005; Logist et al., 2009; Li et al., 2009).

3.2 Grid Search Approach

The GSA is a technique used to approximate the Pareto domain that involves the construction of a grid in the input space of an optimization problem. The objective functions are then calculated at the grid points, and the best points are chosen based on the concept of dominance. The search begins from a coarse grid, and is refined at each iteration. The full GSA procedure is described below.

1. Initially a grid is formed in the feasible region of the input space. The number of initial divisions for each input is specified before beginning the GSA, with 5 initial divisions being used for each input in this study.
2. The values of the objective functions are then calculated for each point in the grid. The procedure generates $\prod_{i=1}^{n}(M_i+1)$ points, where M_i equals the number of divisions for input *i*, and *n* equals the total number of inputs in the specified optimization problem.
3. Next all points in the grid are compared to other points in the grid one at a time, and the number of times a given point is dominated by another is determined.
4. Using the domination count, the best points are then selected to determine the range in the input variables for the construction of the next grid. The minimum and maximum values of the input variables, that contain the non-dominated points, are used as the range for the next grid. As the accuracy of the Pareto domain increases with each iteration, the number of divisions is also increased. In this study the number of divisions was increased by 5 at each iteration.
5. Steps 1-4 are repeated, gradually producing a finer grid in the input space. Once the grid interval for an input variable reaches a pre-determined minimum value, the number of divisions is no longer increased for that input variable. Once all of the input variables have reached the pre-determined minimum grid interval, the approximation of the Pareto domain is complete.

4.0 Application of Principal Component Analysis to the Grid Search Approach

When the GSA is used to approximate the Pareto domain, a large number of dominated points can be generated at each iteration, even as the grid interval decreases and the procedure nears convergence. An example of an optimization problem where this is especially important is demonstrated in Figure 2, which shows the input space of the Pareto domain for a problem involving two input variables. The grid search region is also shown in the figure. In this example, the Pareto domain represents only a very small fraction of the grid search region. Decreasing the grid size will lead to a more accurate Pareto domain, but will also produce an increasing number of points in the grid search region that are not part of the Pareto domain. This limitation of the GSA leads to high computation time as each

iteration generates many redundant points for which the objective functions must be calculated.

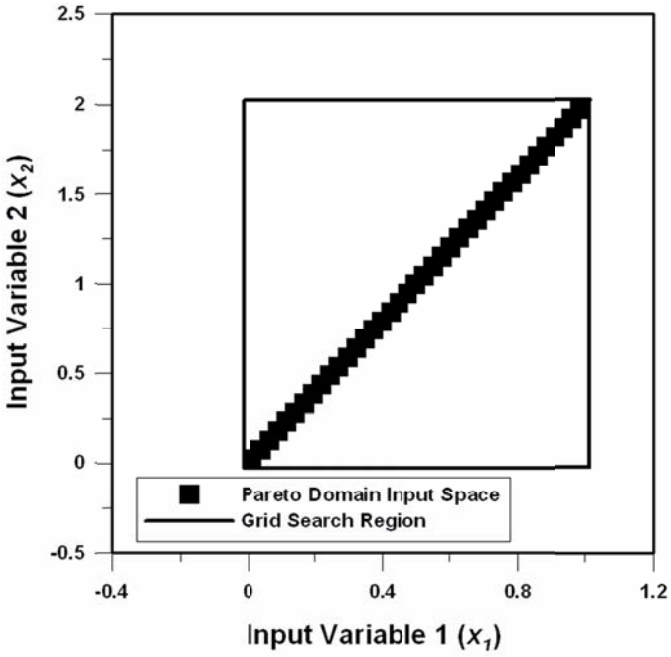

Figure 2: Input space of the Pareto domain for the example problem.

Principal Component Analysis (PCA) is a statistical tool that is most often used to convert a set of possibly correlated variables into a set of uncorrelated variables (Jolliffe, 1986). Principal component analysis has previously been used in MOO techniques (Deb et al., 2005; Brockhoff et al., 2002; Woźniak, 2007) to reduce the complexity of MOO problems by reducing the dimensionality.. In this study, the PCA procedure is used in combination with the GSA, in order to increase the accuracy and efficiency of the GSA. The example in Figure 2 is used below to demonstrate how PCA can benefit the GSA, and the required calculation procedure is described (Jolliffe, 1986).

1. The first step in the procedure is to calculate the covariance matrix. The data set used for this calculation is the input space of the Pareto domain, i.e. the set of x_1 and x_2 values for the example problem. This calculation is shown in Equation (1).

$$\begin{bmatrix} Cov(x_1,x_1) & Cov(x_1,x_2) \\ Cov(x_2,x_1) & Cov(x_2,x_2) \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{N} \frac{\left(x_{1,j}-\bar{x}_1\right)^2}{N-1} & \sum_{j=1}^{N} \frac{\left(x_{1,j}-\bar{x}_1\right)\left(x_{2,j}-\bar{x}_2\right)}{N-1} \\ \sum_{j=1}^{N} \frac{\left(x_{2,j}-\bar{x}_2\right)\left(x_{1,j}-\bar{x}_1\right)}{N-1} & \sum_{j=1}^{N} \frac{\left(x_{2,j}-\bar{x}_2\right)^2}{N-1} \end{bmatrix} \quad (1)$$

where N is equal to the number of points in the input space and $\bar{x}_1$ and $\bar{x}_2$ are the average values of each input variable.

2. Next the eigenvalues are calculated. All optimization problems in this study consisted of two input variables. When two input variables are used, Equation (2) gives the eigenvalues of the dataset (λ_A and λ_B) using the quadratic formula.

$$\lambda_A = \frac{T + (T^2 - 4D)^{0.5}}{2}$$
$$\lambda_B = \frac{T - (T^2 - 4D)^{0.5}}{2} \quad (2)$$

where T is the trace of the covariance matrix, and D is the determinant of the covariance matrix.

3. Next the eigenvectors are calculated. The definition of an Eigenvector V (with elements v_1 and v_2) corresponding to an eigenvalue λ for a system with two input variables is shown in Equation (3).

$$\begin{bmatrix} Cov(x_1,x_1) & Cov(x_2,x_1) \\ Cov(x_1,x_2) & Cov(x_2,x_2) \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} = \lambda \begin{bmatrix} v_1 \\ v_2 \end{bmatrix} \quad (3)$$

Equation (3) always yields two dependent equations for the Eigenvector V. The process of finding vector V can therefore be simplified by setting v_2 to 1, and solving for v_1 from Equation (4).

$$v_1 = \frac{Cov(x_2,x_1)}{\lambda - Cov(x_1,x_1)} \quad \text{or} \quad v_1 = \frac{\lambda - Cov(x_2,x_2)}{Cov(x_1,x_2)} \quad (4)$$

The calculation described by Equations 3 and 4 is performed for both eigenvalues λ_A and λ_B, corresponding to the two eigenvectors V_A and V_B, each with distinct values for elements v_1 and v_2.

4. The principal component projection of the data set is then calculated. This calculation consists of multiplying the data set by the matrix composed of the eigenvectors V_A and V_B.

The calculation procedure outlined in steps 1-4 was performed on the data set shown in Figure 2. The projected dataset is shown in Figure 3.

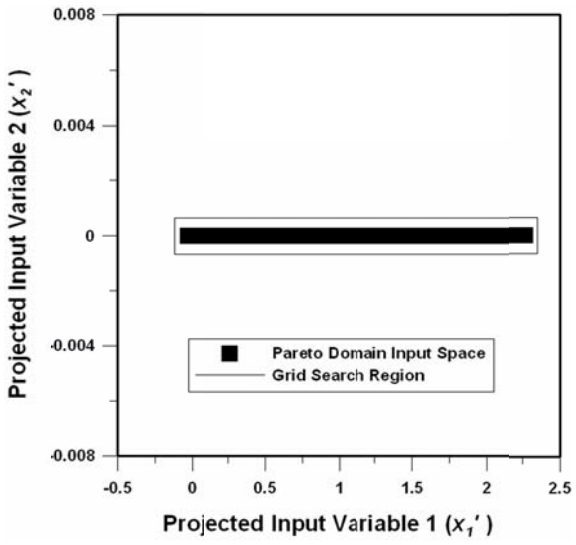

Figure 3: Principal component projection of the input variable space for the example problem.

In Figure 3, the data set is certainly ideal for the application of a grid. For this example the grid search region includes only a small fraction of redundant points, and the dataset has essentially been reduced from a two-variable system into a one-variable system. For a given optimization problem the number of dimensions may not be reduced, but unlike previous MOO studies involving PCA a significant reduction in the search space may still be obtained. This reduction in the search space demonstrates the benefit of projecting the input space using PCA before each grid is produced following the first standard grid search.

Once the grid has been formed it must be projected back to the original frame of reference. This calculation consists of multiplying the grid by the inverse of the eigenvector matrix. The resulting grid for this problem is shown in Figure 4. This figure demonstrates a final resulting grid with a much smaller grid search region, and a smaller fraction of redundant points relative to the standard grid shown in Figure 2.

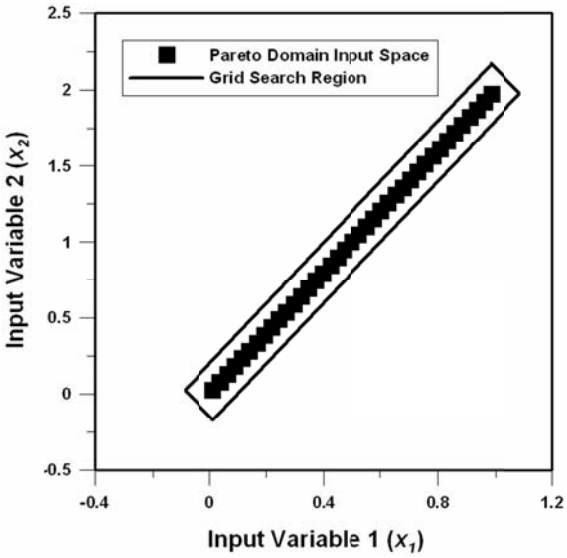

Figure 4: Final grid generated using the PCA calculation procedure for the example problem.

4.1 Principal Component Grid Algorithm

The general procedure for the proposed algorithm, the Principal Component Grid Algorithm (PCGA), is shown below.

1. Initially a small number of iterations are conducted strictly using the procedure discussed in Section 3.0 to generate a rough estimate of the Pareto domain. In this study, two iterations of the original GSA were used before PCA was implemented.
2. The next iteration begins by determining the Eigenvalues and Eigenvectors for the input variable space. Only the non-dominated points are used in this calculation as they represent the most current approximation of the Pareto domain.
3. The principal component projection is then calculated for the input-variable space, and a grid is formed with the projected data set. The grid is then projected back to the original frame of reference, and the corresponding objective functions are evaluated.

As was described in Section 3.2, the number of divisions is increased at each iteration, and the calculation continues until all of the input variables have reached the pre-determined minimum grid size.

5.0 Optimization Problems

Four test problems were used in this study to examine the performance of the PCGA.

5.1 Problem 1

The first optimization problem is a simple system comprised of two input variables and two output functions as shown in Equations (5) and (6). Both objective functions are to be minimized (Shim et al., 2002).

$$f_1(x_1, x_2) = x_1^2 + x_2^2 \tag{5}$$

$$f_2(x_1, x_2) = (x_1 - 1)^2 + (x_2^2 - 2)^2 \tag{6}$$

$-100 < x_1 < 100$ and $-100 < x_2 < 100$

5.2 Problem 2

The next optimization problem described by Equations (7)-(9) involves three output functions and two input variables. All three output functions are to be minimized (Viennet et al., 1995).

$$f_1(x_1, x_2) = 0.5(x_1^2 + x_2^2) + \sin(x_1^2 + x_2^2) \tag{7}$$

$$f_2(x_1, x_2) = \frac{(3x_1 - 2x_2 + 4)^2}{8} + \frac{(x_1 - x_2 + 1)^2}{27} + 15 \tag{8}$$

$$f_3(x_1, x_2) = \frac{1}{x_1^2 + x_2^2 + 1} - 1.1 \exp(-(x_1^2 + x_2^2)) \tag{9}$$

$-3 < x_1 < 3$ and $-3 < x_2 < 3$

5.3 Gluconic Acid (Problem 3)

The third problem is concerned with the production of gluconic acid from glucose using the microorganism *Pseudomonas ovalis*. The overall reaction can be described as follows.

Cells + Glucose + Oxygen $\longrightarrow$ More Cells

Glucose + Oxygen $\xrightarrow{\text{Cells}}$ Gluconolactone

Gluconolactone + Water $\longrightarrow$ Gluconic Acid

The dynamic model of this fermentation process, given in Equations (10)-(14), was developed by Ghose and Ghosh (1971) and accounts for the changes in the concentration of

cells (X), gluconic acid (P), gluconolactone (l), glucose substrate (S), and dissolved oxygen (C). The parameters for the model are given in Table 1.

$$\frac{dX}{dt} = \mu_m \frac{SC}{k_s C + k_0 S + SC} X \tag{10}$$

$$\frac{dP}{dt} = k_p l \tag{11}$$

$$\frac{dl}{dt} = v_l \frac{S}{k_l + S} X - 0.91 k_p l \tag{12}$$

$$\frac{dS}{dt} = -\frac{1}{y_s} \mu_m \frac{SC}{k_s C + k_0 S + SC} X - 1.011 v_l \frac{S}{k_l + S} X \tag{13}$$

$$\frac{dC}{dt} = K_L a (C^* - C) - \frac{1}{y_0} \mu_m \frac{SC}{k_s C + k_0 S + SC} X - 0.09 v_l \frac{S}{k_l + S} X \tag{14}$$

Table 1: Parameters used in the gluconic acid production model.

Parameter	Value	Unit
μ_m	0.39	h^{-1}
k_s	2.5	g/L
k_0	0.00055	g/L
k_P	0.645	h^{-1}
v_l	8.3	mg/UOD h
K_l	12.8	g/L
Y_s	0.375	UOD/mg
Y_0	0.89	UOD/mg
C^*	0.00685	g/L
X_0 (initial cell concentration)	1	UOD/mL
S_0 (initial substrate concentration)	50	g/L

For this model, there exist a large number of input and output variables. In this study, two input variables, the batch time t_B (x_1) and the overall oxygen mass transfer coefficient $K_L a$

(x_2), were varied in the optimization, and all other potential input variables remained constant. Two output variables, the productivity P_f/t_B (f_1) and the final concentration of gluconic acid P_f (f_2) were retained (Thibault, 2009) such that the optimization problem is a two-input and two-output system. The lower and upper bounds for the two input variables are $5 < t_B < 15$ h and $50 < K_La < 300$ h^{-1}.

5.4 PI Controller (Problem 4)

The final problem is concerned with the control of a process that can be represented by a First-Order Plus Dead-Time (FOPDT) model, subject to a unit step change in the controlled variable. A FOPDT process transfer function is frequently used for controller optimization, as it adequately represents a large number of higher-order systems and industrial processes (Cvejn 2009; Madhuranthakam et al., 2008; Roy et al., 2005). The process can be represented in the Laplace domain by Equation (15) (Marlin, 2000).

$$\frac{y_t}{u_t} = \frac{K_p e^{-\theta s}}{\tau s + 1} \tag{15}$$

Equation (15) was solved numerically using finite differences. The resulting expression is shown in Equation (16).

$$y_t = y_{t-\Delta t} + \frac{\Delta t}{\tau}\left[K_p u_{t-\Delta t-\theta} - y_{t-\Delta t} \right] \tag{16}$$

where u_t represents the manipulated variable at time t, K_p represents the process gain, τ represents the process time constant, Δt represents the small integration time step used in the calculation, y_t represents the controlled variable at time t and θ represents the time delay.

Despite tremendous advances in process control and the development of numerous control algorithms, the PI controller still remains the most commonly used control algorithm in industrial applications (Desbourough and Miller, 2002; Lee et al., 2010; Koo et al., 2001; Tan et al., 2001). The reason for its widespread industrial utilization is its simplicity and ease of implementation. In addition, when significant dead time is present the derivative term in a PID controller leads to an incorrect response such that a PI controller is favored (Koo et al., 2001; Tan et al., 2001). Therefore a PI controller was used to control the FOPDT process in

this optimization study. The equation describing the response of the manipulated variable is shown in Equation (17) with a bias term of zero (Marlin, 2000).

$$u_t = K_c \varepsilon_t + \frac{K_c}{\tau_I} \int_0^t \varepsilon_t dt \tag{17}$$

Equation (17) is again solved numerically using finite differences. The resulting expression is shown in Equation (18) which represents the velocity form of the PI Controller.

$$u_t = u_{t-\Delta t} + K_c \left(\varepsilon_t - \varepsilon_{t-\Delta t} \right) + \frac{K_c}{\tau_I} \varepsilon_t \Delta t \tag{18}$$

Where ε_t is the error at time t. K_c is the controller gain, and τ_I represents the integral time.

Although several variables affect the controller performance, only the integral time τ_I and the controller gain K_c can be varied when tuning the controller. These two input variables were therefore used in this optimization problem. The feasible region for each input variable was set between 0.1 and 10. The dead time, process gain and time constant were set to 5s, 1, and 5s respectively, and were not varied in the optimization procedure.

For the tuning of a PI controller, many performance criteria can be used alone or as a weighted sum of two or more. In this investigation, a multi-objective optimization is performed for a FOPDT process subject to a unit set-point change in the controlled variable. Three controller performance criteria were considered in the optimization: the Integral of the Time Weighted Absolute Error (ITAE), the Integral of the Squares of the Differences in the Manipulated Variable (ISDU) and the settling time. The mathematical expressions of ITAE and ISDU are given in Equations (5) and (6), respectively. The ITAE measures the cumulative deviation of the controlled variable from the set point, and penalizes deviations that are not resolved in a short period of time. ISDU measures the changes in the manipulated variable and favours a smooth response (Seborg, 2004).

$$ITAE = \int_0^{t_{final}} t\left|\varepsilon_t\right| dt \simeq \sum_{k=1}^{t_{final}/\Delta t} t\left|\varepsilon_k\right| \Delta t \tag{19}$$

$$ISDU = \int_0^{t_{final}} \Delta u_t^{\,2} dt \simeq \sum_{k=1}^{t_{final}/\Delta t} \left(u_k - u_{k-1}\right)^2 \Delta t \qquad (20)$$

The settling time is defined in this investigation as the time that the process takes to stabilize within ± 5% of the final steady state value. The settling time for a typical response from a unit step change in the controlled variable is demonstrated in Figure 5.

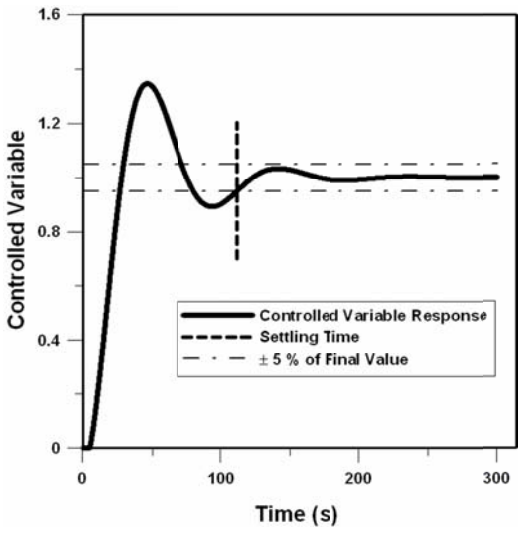

Figure 5: Graphical illustration of the settling time for a typical response.

6.0 Results and Discussion

For the four illustrative examples, the Pareto domain was approximated using the GSA, PCGA and NSGA-II. For both grid methods, the calculation continued until the grid size for each input variable was equal to 0.05. For the NSGA-II Pareto domains, the algorithm was applied such that the same final number of points was obtained as in the PCGA. It should be noted that due to the inherent variability of NSGA-II, all results shown for this algorithm are based on an average of six runs. All computer simulations were performed with a 4.5 GHz Intel Pentium 4 processor with 1.49 GB of RAM.

After approximating the Pareto domains for each optimization problem, they were inspected visually. The visual inspection allows examining how evenly each algorithm spanned the Pareto domain, and the relative accuracy of each algorithm can be estimated. As an example, the Pareto domain for the PI controller (Problem 4) is shown in Figures 6-8 as approximated respectively by each of the three algorithms.

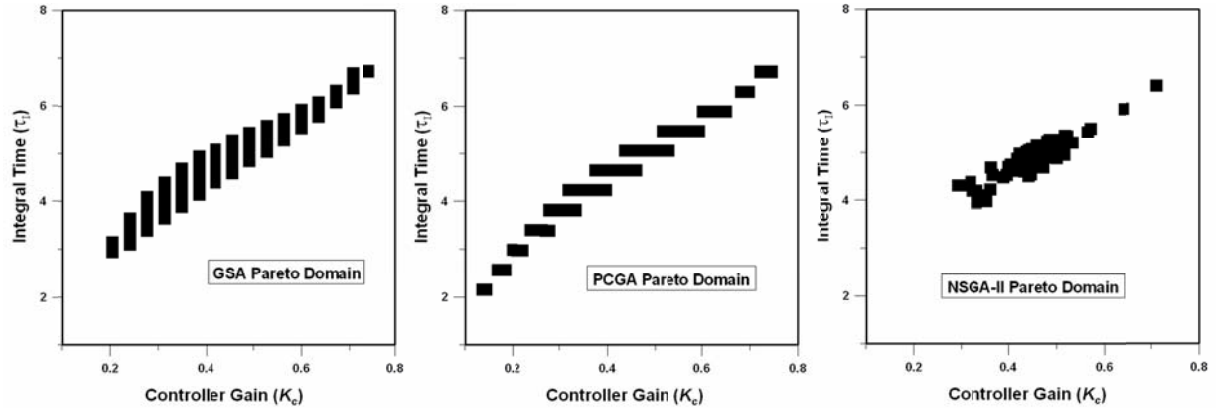

Figure 6: Input space of the Pareto domain for Problem 4 using GSA, PCGA and NSGA-II.

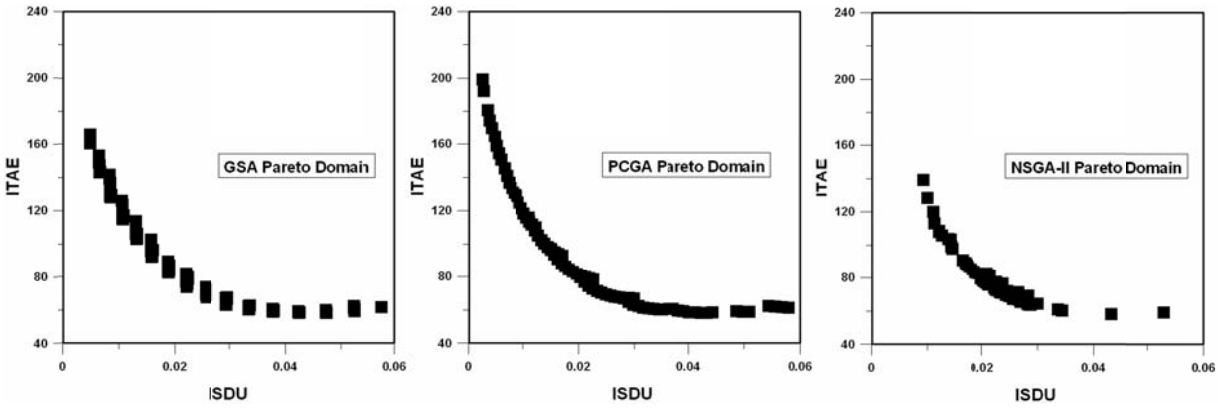

Figure 7: ITAE versus ISDU for the Pareto domain of Problem 4 using GSA, PCGA and NSGA-II.

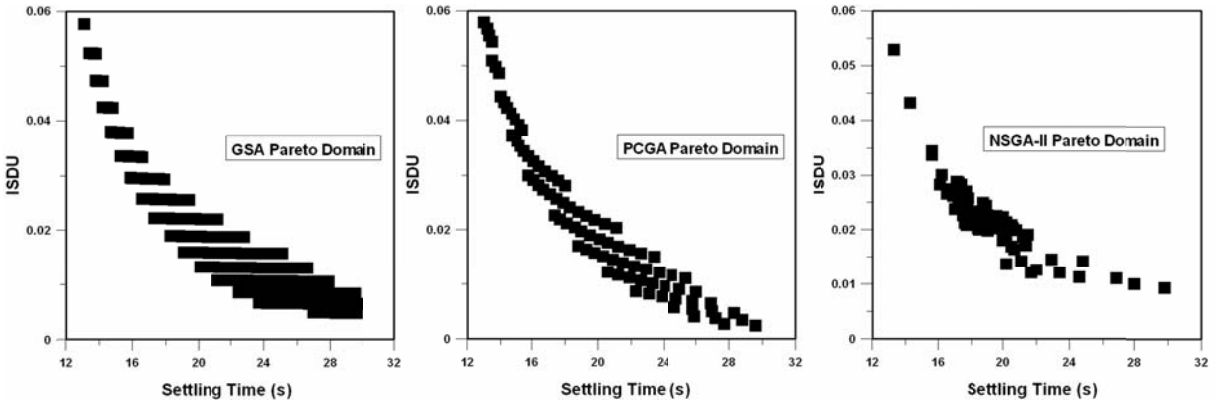

Figure 8: ISDU versus settling time for the Pareto domain of Problem 4 using GSA, PCGA and NSGA-II.

Results of Figures 6-8 clearly show that the PCGA and the GSA spanned the Pareto domain more evenly and completely than NSGA-II. This trend was observed for all four optimization problems. When the relative accuracy of the three algorithms is considered, they appear to be similar when inspected visually. This observation was again found for all optimization problems studied. Therefore, to assess the accuracy of the three algorithms a more rigorous comparison method is necessary.

One may consider evaluating the accuracy of the Pareto domain approximation by calculating the distance of each point from the true Pareto domain. However, this method is not possible for many engineering problems (such as Problems 3 and 4 in this study) as the true Pareto domain cannot be determined. Instead, the relative accuracy of each of the algorithms was evaluated in this study. This was done by combining the Pareto domains obtained from the two algorithms that were being compared. Initially the two populations consist strictly of non-dominated points. However, after the two populations are combined and the dominance test is performed on the combined population, some of the points may become dominated. If, for example, when PCGA is compared to GSA it is found that the majority of the new dominated points originated from the GSA population, then it can be stated that the accuracy of the GSA is lower than that of the PCGA. This relative dominance test was performed on the combined populations obtained with the different methods for each optimization problem studied. The percentage of dominated points found in each population, for each optimization problem is shown in Figure 9 for the combined populations.

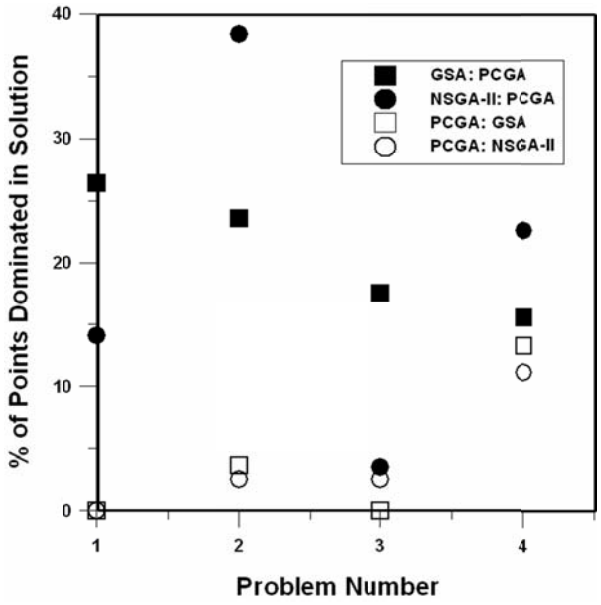

Figure 9: Percentage of points that were dominated for the GSA and NSGA-II Pareto domains relative to the PCGA (dark symbols), and vice versa (open symbols).

The results of Figure 9 show that for all optimization problems studied the PCGA produced a Pareto domain with higher accuracy. For all problems, the percentage of dominated points was smaller in the PCGA Pareto domains compared to the GSA and NGSA-II Pareto domains. For Problems 1 and 3 the greatest percentage of dominated points, corresponding to the lowest accuracy, was found in the GSA Pareto domains, whereas for Problems 2 and 4 the lowest accuracy was found in the NGSA-II Pareto domains. Finally for Problems 1 and 2, a significantly larger percentage of dominated points were found in the GSA and NSGA-II solutions relative to the PCGA, whereas this difference was less significant for Problems 3 and 4.

The performance of each algorithm in approximating the Pareto domain was further evaluated by examining the computation time and the number of objective function calls required in approximating the Pareto domain for each optimization problem. The results of this comparison are presented in Figure 10.

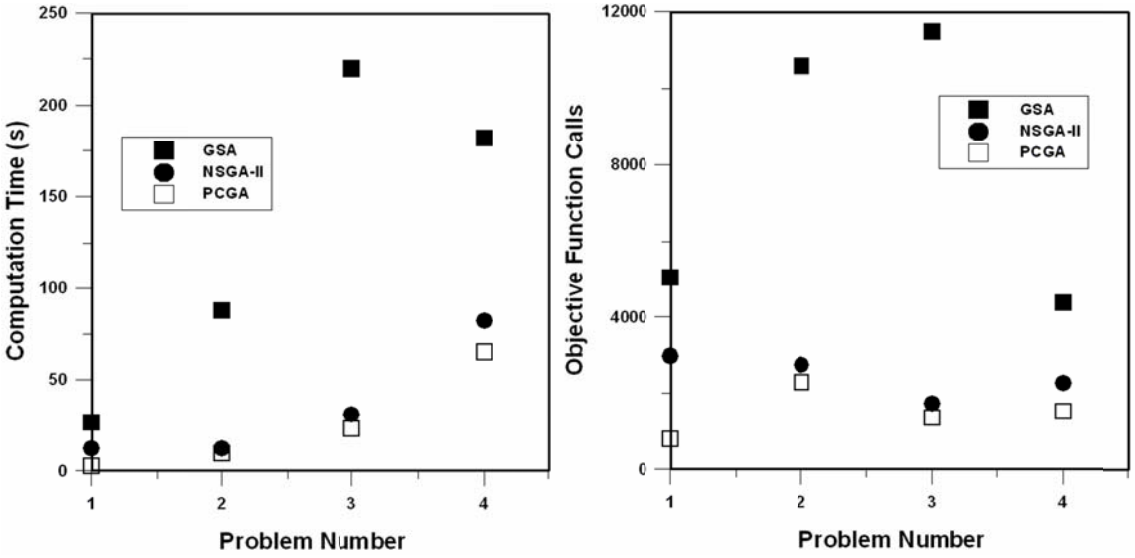

Figure 10: Computation time and objective function calls for the PCGA, GSA and NSGA-II for the four optimization problems.

Figure 10 shows that the PCGA approximated the Pareto domain with both lower computation time, and fewer objective function calls than both GSA and NSGA-II for all optimization problems. For Problems 2, 3 and 4 it was found that GSA required significantly larger computation time and more objective function calls relative to PCGA, whereas the increase in the computation time and number of objective function calls required for the NGSA-II Pareto domains was less significant. For Problem 1 the computation time and objective function calls were still greatest for the GSA, but the difference between NSGA-II and GSA was less pronounced. From the results shown in Figures 9 and 10 it is clear that PCGA demonstrates a significant improvement from the GSA, producing a Pareto domain with increased accuracy and efficiency relative to both the GSA and NSGA-II.

8.0 Conclusions

In this study, PCA is incorporated into the GSA with the goals of defining the Pareto domain more accurately and efficiently relative to GSA. The performance of the proposed algorithm, the PCGA, was evaluated using four test problems, including a PI controller model for a FOPDT system and a gluconic acid model.

For all problems tested it was found that the PCGA produced a more accurate Pareto domain than both the GSA and NSGA-II, and spanned the Pareto domain more evenly and

completely. It was also found that the PCGA approximated the Pareto domain for each optimization problem with the highest efficiency of all three algorithms, with a reduction in both the computation time and the number of objective function calls required in approximating the Pareto domain. This study indicates that PCGA is an effective and efficient algorithm for multi-objective optimization problems.

9.0 Nomenclature

Variable	Definition	Unit
Cov	Element in Covariance Matrix	dimensionless
D	Determinant	dimensionless
f	Objective function in optimization problem	varies
$ISDU$	Integral of the Squares of the Differences in the Manipulated Variable	varies
$ITAE$	Integral of the Time Weighted Absolute Error	varies
K_c	Controller Gain	units of u/y
K_p	Process Gain	units of y/u
m	Number of objective criteria	dimensionless
M	Number of divisions in PCGA grid	dimensionless
N	Number of points in Pareto domain	dimensionless
n	Number of input variables	dimensionless
P	Point in the Pareto domain	dimensionless
T	Trace	dimensionless
u	Manipulated variable	varies
V	Eigenvector	dimensionless
x	Input function in optimization problem	varies
$\overline{x}$	Average input function value in Pareto domain	varies

y	Controlled variable	varies
Greek Symbols		
Δt	Time step	s
ε	Error	varies
λ	Eigenvalue	dimensionless
θ	Dead time	s
τ	Time constant	s
τ_I	Integral Time	s

Subscript	Definition
t	Time
j	Point in the Pareto domain

10.0 References

Agarwal, A. and Gupta, S.K., 2008. Jumping gene adaptations of NSGA-II and their use in the multi-objective optimal design of shell and tube heat exchangers. *Chemical Engineering Research and Design*, 86 (2), 123-139.

Brockhoff, D. and Zitzler E., 2007. Dimensionality Reduction in Multiobjective Optimization: The Minimum Objective Subset Problem. *In*: K. Waldmann and U. Stocker, ed. *Annual International Conference of the German Operations Research Society,* 6-8 September 2006 Karlsruhe. Germany: Springer, 423-429.

Cvejn, J., 2009. Sub-optimal PID controller settings for FOPDT systems with long dead time. *Journal of Process Control,* 19 (9), 1486-1495.

Deb, K., 2001. *Multi-objective optimization using Evolutionary Algorithms*. England: John Wiley & Sons, Ltd.

Deb, K., Pratap, A., Agarwal, S. and Meyarivan, T., 2002. A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6 (2), 182-197.

Deb, K. and Saxena, D., 2005. On Finding Pareto-Optimal Solutions Through Dimensionality Reduction for Certain Large-Dimensional Multi-Objective Optimization Problems. *Kanpur Genetic Algorithms Laboratory,* Report 2005011, 1-19.

Desbourough, L. and Miller, R, 2001. Increasing customer value of industrial control performance monitoring Honeywell's experience. *In*: F. Rawlings, B. Ogunnaike and J. Eaton, ed. *Proceedings of the 6th International Conference on Chemical Process Control*, January 2001 USA: AIChE, 172-192.

Fonteix, C., Massebeuf. S., Pla, F. and Kiss, L., 2004. Multicriteria optimization of an emulsion polymerization process. *European Journal of Operational Research,* 153 (2), 350–359.

Ghose, T. and Ghosh, P., 1976. Kinetic analysis of gluconic acid production by *Pseudomonas ovalis*. *Journal of Applied Chemistry and Biotechnology,* 26 (1), 268-277.

Hägglund, T., 1996. An Industrial Dead-time Compensating PI Controller. *Control Engineering Practice*, 4 (6), 749-756.

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

Joliffe, I.T., 1986. *Principal Component Analysis*. New York: Springer-Verlage.

Koo, D.G., Lee, J., Lee, D.K., Han, C., Gyu, L.S., Jung, J.H. and Lee, M., 2001. A Tuning of the Nonlinear PI Controller and Its Experimental Application. *Korean Journal of Chemical Engineering*, 18 (4), 451-455.

Lee, S. and Park, J.H., 2010. Performance improvement of PI controller with nonlinear error shaping function: IDA-PBC approach. *Applied Mathematics and Computation*, 215 (10), 3620-3630.

Li, C., Zhang, X., Zhang, S. and Suzuki, K., 2009. Environmentally conscious design of chemical processes and products: Multi-optimization method. *Chemical Engineering Research and Design*, 87 (2), 233-243.

Logist, F., Van Erdeghem, P.M.,and Van Impe, J.F., 2009. Efficient deterministic multiple objective optimal control of (bio)chemical processes. *Chemical Engineering Science,* 64 (11), 2527-2538.

Madhuranthakam, C.R., Elkamel, A. and Budman, H., 2008. Optimal tuning of PID controllers for FOPTD, SOPTD and SOPTD with lead processes. *Chemical Engineering and Processing*, 47 (2), 251-264.

Marlin, T., 2001. *Process Control: Designing Processes and Control Systems for Dynamic Performance*. USA: John Wiley & Sons, Ltd.

Mokeddem, D. and Khellaf, A., 2009. Optimal Solutions of Multiproduct Batch Chemical Process Using Multiobjective Genetic Algorithm with Expert Decision System. *Journal of Automated Methods and Management in Chemistry*, 2009, 1-9.

Poloni, C., Giurgevich, A., Onesti, L. and Pediroda, V., 2000. Hybridization of a multi-objective genetic algorithm, a neural network, and a classical optimizer for a complex design problem in fluid dynamics. *Computer Methods in Applied Mechanics and Engineering*, 186 (2-4), 403-420.

Roy, A. and Iqbal, K., 2005. PID controller tuning for the first-order-plus-dead-time process model via Hermite-Biehler theorem. *ISA Transactions,* 44 (3), 363-378.

Seborg, D.E., Edgar, T.F. and Mellichamp D.A., 2004. *Process Dynamics and Control*. USA: John Wiley & Sons, Inc.

Shim, M., Suh, M., Furukawa, R., Yagawa, G. and Yoshimura, S., 2002. Pareto-based continuous evolutionary algorithms for multiobjective optimization. *Engineering Computations*, 19 (1), 22-48.

Tan, K.K., Lee, T.H. and Leu, F.M., 2001. Predictive PI versus Smith control for dead-time compensation. *ISA Transactions*, 40 (1), 17-29.

Tarafder, A., Rangaiah, G.P. and Ray, A.K., 2005. Multiobjective optimization of an industrial styrene monomer manufacturing process. *Chemical Engineering Science,* 60 (2), 347-363.

Thibault, J., 2009. Net Flow and Rough Sets: Two Methods for Ranking the Pareto Domain. *In*: Rangaiah G., Editor, *Advances in Process Systems Engineering – Vol. 1: Multi-Objective Optimization: Techniques and Applications in Chemical Engineering*, World Scientific Publishing, Singapore, 189-236.

Viennet, R., Fonteix, C. and Marc, I., 1996. Multicriteria optimization using a genetic algorithm for determining a Pareto set. *International Journal of System Science*, 27 (2), 255-260.

Woźniak, P., 2007. Dimensionality reduction in evolutionary multiobjective design: case study. *In*: D. Thierens et al., editors. *Proceedings of the 9th annual conference on Genetic and Evolutionary Computation,* 7-11 July 2007 London. UK: Association for Computing Machinery, Inc, 913-915.

Chapter 4

New PI Controller Tuning Methods using Multi-Objective Optimization

Allan Vandervoort, Jules Thibault* and Yash Gupta

Abstract

In this paper, new tuning methods are developed for a PI controller for processes represented by a First-Order Plus Dead Time (FOPDT) transfer function. The developed methods involve approximating the Pareto domain associated with the minimization of three performance criteria. Two tuning methods were developed, achieving optimal controller performance by specifying either one of the controller input parameters or the desired values of the performance criteria. The developed controller tuning methods were compared to several previously developed controller correlations. Finally, the tuning methods were applied to a fourth order process subjected to a set point change and a disturbance, and demonstrated excellent performance.

Keywords: PI controller, Tuning, FOPDT, Pareto domain, multi-objective optimization

Journal Publication: Canadian Journal of Chemical Engineering
Publisher: Wiley on behalf of the Canadian Society of Chemical Engineering
Publication Status: Submitted

1.0 Introduction

Efficient process control is an essential element in ensuring industrial chemical plants operate economically and optimally in a safe manner, while meeting product specifications and environmental regulations (Stephanopoulos, 1984). Despite tremendous advances in process control and the development of numerous control algorithms, the PI controller still remains the most commonly used control algorithm in industrial applications (Desbourough and Miller, 2002; Lee et al., 2010; Koo et al., 2001; Tan et al., 2001). The reason for its widespread industrial utilization is its simplicity and ease of implementation. In addition, when significant dead time is present the derivative term in a PID controller leads to an incorrect response such that a PI controller is favored (Koo et al., 2001; Tan et al., 2001). The development of efficient and robust tuning methods for PI controllers is therefore very important.

A properly configured controller for a chemical process should be robust, minimize excessive controller action, and produce a stable response with no final offset (Seborg et al., 2004). Many controller correlations have been developed for tuning PI controllers such as those proposed by Ziegler and Nichols (1942), Cohen and Coon (1953), Chien and Fruehauf (1990), and Skogestad (2003). Although these and many other correlations have been implemented in process control systems, no controller correlation can achieve all of the desired performance criteria simultaneously because they inherently involve conflicts and trade-offs (Seborg et al., 2004).

In recent years, the multi-objective optimization of PI controllers has been studied in an attempt to better understand the trade-off between various controller objectives. In many of these studies, multi-objective optimization is performed for the control of specific processes (Fonseca, 1996; Xue et al., 2010). In other studies, general tuning rules have been developed by considering multiple objectives (Kookos, 1999; Tavakoli et al., 2009) but in these methods only one set of controller parameters was suggested for a given process. Although the trade-offs between each performance objective were considered by the authors during the development of these tuning methods, they cannot be considered by the decision maker when the tuning methods are applied to industrial control systems. In this investigation, new PI controller tuning methods based on multi-objective optimization are proposed. The methods

in this study, where multiple objective criteria are simultaneously optimized, were developed to improve the decision maker's understanding of the trade-off associated with each objective, before optimum controller parameters are chosen.

2.0 PI Controller Model

In this study, a First Order Plus Dead Time (FOPDT) process model was used for the development of the PI controller tuning methods. A FOPDT process transfer function is frequently used for controller optimization, as it adequately represents a large number of higher-order systems and industrial processes (Cvejn, 2009; Madhuranthakam et al., 2008; Roy et al., 2005). A FOPDT process can be represented in the Laplace domain by Equation (1) (Marlin, 2000).

$$\frac{y_t}{u_t} = \frac{K_p e^{-\theta s}}{\tau s + 1} \tag{1}$$

Equation (1) is solved numerically using finite differences. The resulting expression is shown in Equation (2).

$$y_t = y_{t-\Delta t} + \frac{\Delta t}{\tau}\left[K_p u_{t-\Delta t-\theta} - y_{t-\Delta t} \right] \tag{2}$$

where u_t represents the manipulated variable at time t, K_p represents the process gain, τ represents the process time constant, Δt represents the small integration time step used in the calculation, y_t represents the controlled variable at time t, and θ represents the time delay.

For a PI controller the equation describing the response of the manipulated variable is shown in Equation (3) (Marlin, 2000) for a bias term of zero.

$$u_t = K_c \varepsilon_t + \frac{K_c}{\tau_I} \int_0^t \varepsilon_t \, dt \tag{3}$$

Equation (3) is again solved numerically using finite differences as shown in Equation (4), which represents the velocity form of the PI controller.

$$u_t = u_{t-\Delta t} + K_c \left(\varepsilon_t - \varepsilon_{t-\Delta t} \right) + \frac{K_c}{\tau_I} \varepsilon_t \Delta t \tag{4}$$

Where ε_t is the error at time t, K_c is the controller gain, and τ_I is the integral time.

3.0 Approximating the Pareto Domain

3.1 Pareto Domain

The Pareto domain consists of all non-dominated solutions within the feasible solution domain. For a general definition of dominance two points are considered, P_1 and P_2. Each point is comprised of n input variables ($x_1, x_2, x_3 \ldots x_n$) and m output (objective or performance criteria) values ($f_1, f_2, f_3 \ldots f_m$). For a point P_1 to dominate a point P_2, the following two conditions must hold (Deb, 2001):

- None of the criteria values, f_1 to f_m, for P_1 are worse than the corresponding criteria values, f_1 to f_m, for P_2. For example, if all output criteria values are to be maximized then no criteria values in P_1 can be smaller than the corresponding criteria value for P_2.
- At least one objective criterion for P_1 must be better than the corresponding objective criterion for P_2.

If P_1 dominates P_2 then P_2 is a dominated point and if P_1 does not dominate P_2 and P_2 does not dominate P_1, then both points are non-dominated points with respect to each other.

For a given optimization problem, only points within the Pareto domain are considered when choosing ideal operating conditions, since all points outside of the Pareto domain are dominated points and are therefore worse for all criteria than a particular point in the Pareto domain. The Pareto domain also allows the decision maker to understand the trade-off between each objective before choosing ideal operating conditions.

3.2 The Principal Component Grid Algorithm

In this study, the Principal Component Grid Algorithm (PCGA) developed by Vandervoort (2010) was used to approximate the Pareto domain. The PCGA is a grid-based algorithm that incorporates Principal Component Analysis (PCA). Using PCA increases the accuracy and efficiency of the Pareto domain approximation by reducing the size of the grid search region. It was shown by Vandervoort (2010) that for all cases studied, the PCGA approximated the Pareto domain with higher accuracy than both the standard grid procedure (Halsall-Whitney

et al., 2006) and the Non-Dominated Sorting Genetic Algorithm II (NSGA-II) (Deb et al., 2002) which is commonly used in MOO studies.

The procedure for the Principal Component Grid Algorithm (PCGA) is briefly described as follows:

1. Initially a small number of iterations are conducted strictly using a standard grid search to generate a rough estimate of the Pareto domain. A grid is first constructed in the input space of an optimization problem. The objective functions are then calculated at the grid points, and the best points are retained based on the concept of dominance.
2. The next iteration begins by determining the Eigenvalues and Eigenvectors for the input variable space. Only the non-dominated points are used in this calculation as they represent the most current approximation of the Pareto domain.
3. The principal component projection is then calculated for the input-variable space, and a grid is formed with the projected data set. The grid is then projected back to the original frame of reference, the corresponding objective functions are evaluated and the best points are again retained based on the concept of dominance.
4. Steps 2 and 3 are repeated, decreasing the number of divisions in the grid at each iteration as the accuracy of the Pareto domain increases. The calculation continues until all of the input variables have reached a pre-determined minimum grid size.

3.3 Optimization Problem

Although several variables affect the controller performance, only the integral time τ_I and the controller gain K_c can be varied when tuning a PI controller. In this study, relative controller parameters were used to ensure that the optimization results could be generalized for all possible FOPDT processes. Specifically, the relative controller gain ($K_c\,K_p$) and relative integral time (τ_I/τ) were used as input variables. The feasible region for each input variable was set between 0.1 and 10.

For the tuning of a PI controller, many performance criteria can be used alone or as a weighted sum of two or more. In this investigation, a multi-objective optimization is performed for a FOPDT process subject to a unit set point change in the controlled variable.

Three controller performance criteria were considered in the optimization: the Integral of the Time Weighted Absolute Error (ITAE), the Integral of the Squares of the Differences in the Manipulated Variable (ISDU) and the settling time. The mathematical expressions of ITAE and ISDU are given in Equations (5) and (6), respectively. The ITAE measures the cumulative deviation of the controlled variable from the set point, and penalizes deviations that are not resolved in a short period of time. ISDU measures the changes in the manipulated variable and favours a smooth response (Seborg, 2004).

$$ITAE = \int_0^{t_{final}} t|\varepsilon_t|dt \simeq \sum_{k=1}^{t_{final}/\Delta t} t|\varepsilon_k|\Delta t \tag{5}$$

$$ISDU = \int_0^{t_{final}} \Delta u_t^{\,2} dt \simeq \sum_{k=1}^{t_{final}/\Delta t} \left(u_k - u_{k-1}\right)^2 \Delta t \tag{6}$$

The settling time is defined in this investigation as the time that the process takes to stabilize within ± 5% of the final steady state value. The settling time for a typical response from a unit step change in the controlled variable is demonstrated in Figure 1.

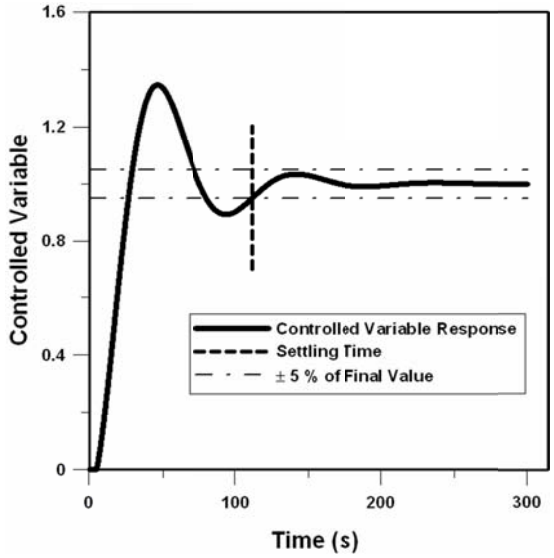

Figure 1: Graphical illustration of the settling time for a typical response.

To further ensure that the controller optimization results can be generalized for all possible FOPDT processes, relative objective criteria were chosen for this optimization. The

objectives chosen were the relative ITAE (ITAE/τ^2), the relative settling time (t_{set}/τ) and the ISDU. The ISDU objective did not directly depend on the response time of the controller and was not placed in relative form because the tuning methods in this study were developed for a unit step set point change. The optimization was performed for varying values of the relative dead time (θ/τ). It was found that regardless of the specific values of the dead time, time constant and process gain, each value of the relative dead time led to one unique Pareto domain. For example, the Pareto domain for a time constant of 100 s, a dead time of 20 s and any value of the process gain was identical to the Pareto domain for a time constant of 5 s, a dead time of 1 s and any value of the process gain when the relative variables were considered. The optimization problem is shown schematically in Figure 2.

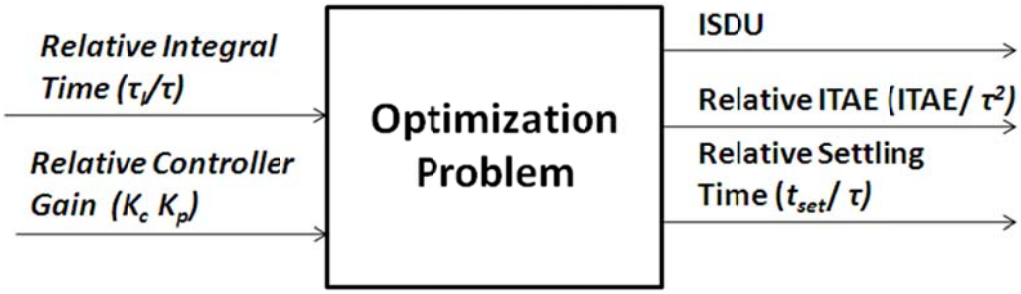

Figure 2: Optimization problem showing input variables and objective functions.

4.0 Optimization Results

The Pareto domain for the generalized controller model, subject to a unit step change in the set point was approximated using the PCGA. The approximation was performed until the grid interval for both input variables was equal to 0.05. Multiple Pareto domains were approximated by varying the value of the relative dead time. Figure 3 shows the Pareto domains for varying values of the relative dead time, with Pareto domains corresponding to a relative dead-time of 0.2 to 2 shown.

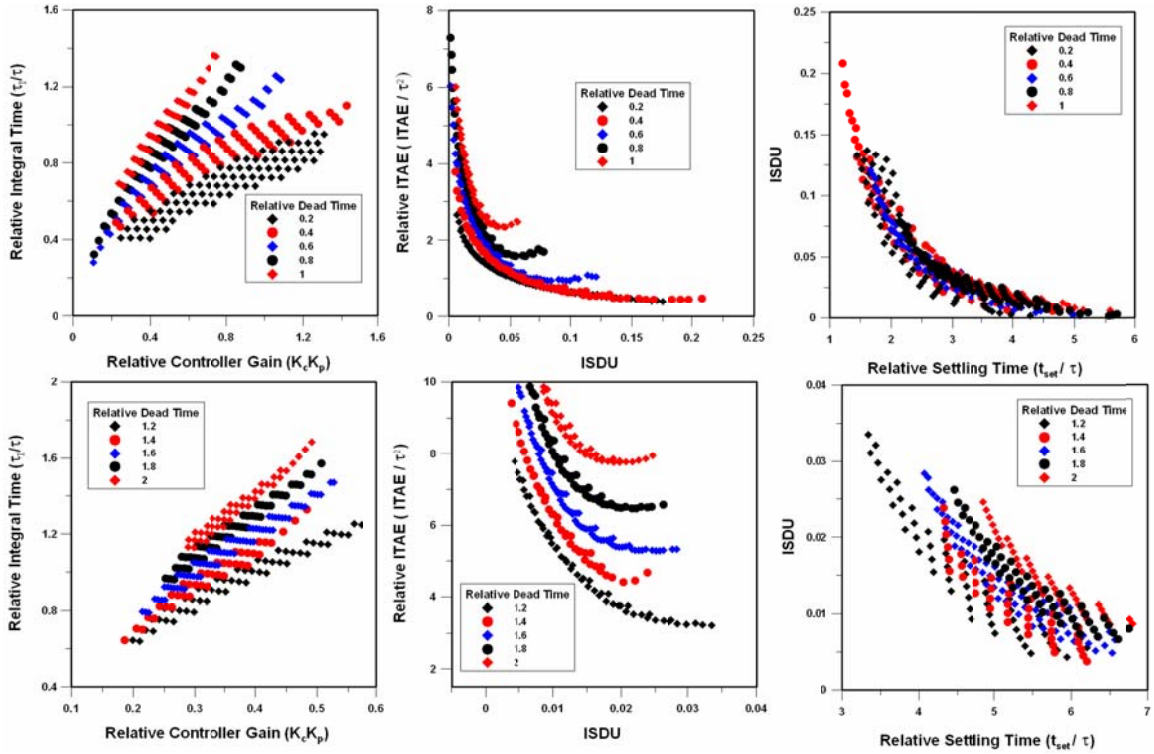

Figure 3: Pareto domains for the generalized controller model with varying values of the relative dead time.

Figure 3 shows that the relative dead time has a significant effect on the input space of the generalized controller Pareto domain. Increasing the relative dead time leads to an increase in the slope of the Pareto domain input space and to changes in the range of the input variables defining the Pareto domain. Despite these variations, the input space of each Pareto domain is visually very narrow and follows closely a straight line. This is an important observation which suggests that if the relative controller gain is chosen, only a very narrow range in the relative integral time will lead to optimum controller performance, and vice versa. The strong correlation between the two controller parameters implies that when configuring a PI controller, only one of the two parameters needs to be specified, as the other can be obtained based on this strong correlation. Any selection of controller parameters outside of the narrow bands shown in Figure 3 will lead to a dominated point, which corresponds to deteriorated controller performance as all three performance criteria would be worse than a particular point within the Pareto domain.

Figure 3 also shows the effect of changing the relative dead time on the objective functions. It is apparent that both the relative ITAE and the relative settling time are highest at larger

values of the relative dead time, whereas the ISDU is smallest at larger values of the relative dead time.

5.0 Controller Tuning

5.1 Method 1

Using the Pareto domain, ideal controller parameters can be determined for a given set of process parameters. The first tuning method involves setting one of the controller parameters, the gain or the integral time, such that the corresponding relative parameter falls within the Pareto domain. One Pareto domain was generated for each value of the relative dead time. The ranges of values for both relative controller parameters for each Pareto domain are presented in Table 1. The second controller parameter can then be specified graphically or using the linear relationship between the two relative controller parameters as defined by Equations (7)-(9). The values of both the slope and the intercept for each value of the relative dead time are given in Table 3. Equations (7)-(9), along with the information found in Table 3 allow for the calculation of the integral time for a pre-specified value of the controller gain, and vice versa, to achieve optimum controller performance.

$$\frac{\tau_I}{\tau} = \left[m\left(K_c K_p\right) + b \right] \tag{7}$$

$$K_c K_p = \left(\frac{\tau_I}{\tau} - b \right) \frac{1}{m} \tag{8}$$

$$m = 1.256 \frac{\theta}{\tau} + 0.06280 \tag{9}$$

Table 1: Slope, intercept and input variable ranges for the generalized controller Pareto domain

θ/τ	Slope (m)	Intercept (b)	Optimum Range of K_cK_p	Optimum range of τ_I/τ
0.2	0.325	0.317	0.247-1.30	0.407 - 0.952
0.4	0.479	0.447	0.223-1.42	0.466-1.10
0.6	0.841	0.351	0.106-1.09	0.280-1.26
0.8	1.13	0.335	0.109-0.876	0.321-1.31
1	1.30	0.382	0.236-0.748	0.679-0.748
1.2	1.57	0.373	0.199-0.572	0.638-1.25
1.4	1.87	0.294	0.186-0.483	0.645-1.33
1.6	2.09	0.370	0.214-0.527	0.790-1.47
1.8	2.30	0.392	0.252-0.507	0.961-1.57
2	2.55	0.351	0.289-0.491	1.13-1.68

5.2 Method 2

Method 1 ensures that an optimum set of controller parameters will be obtained, but the values of the objective functions are not specified a priori. An alternative method for configuring a PI controller is to first choose the desired values of the relative objective functions. For this tuning method, a well-defined relationship must exist between each relative objective function and each relative controller parameter, to ensure that once a point is chosen in the objective space the corresponding input variables (controller tuning parameters) can be calculated. To confirm that this relationship does exist, the values of each objective performance criterion were plotted against the values of the two relative controller parameters. These plots are shown in Figures 4 and 5 for different values of the relative dead time. Figures 4 and 5 show that the relationship between each objective performance criterion and each relative controller parameter is well defined, and can therefore be used for controller tuning.

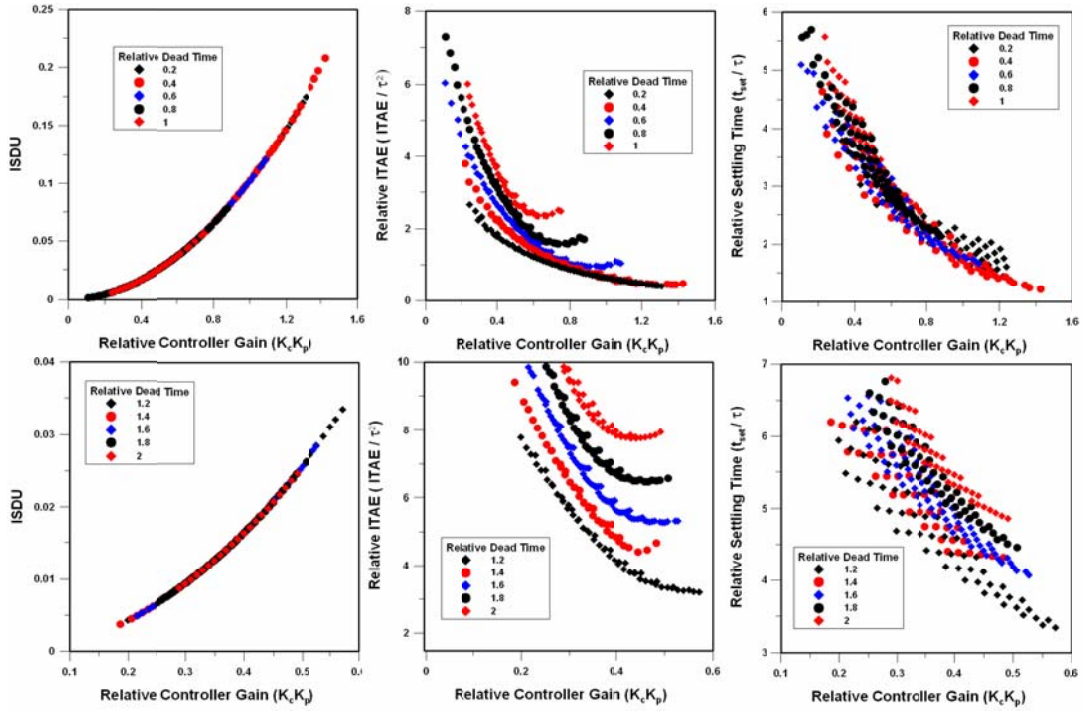

Figure 4: Graph of relative objective functions versus the relative controller gain for the generalized controller model for different relative dead times.

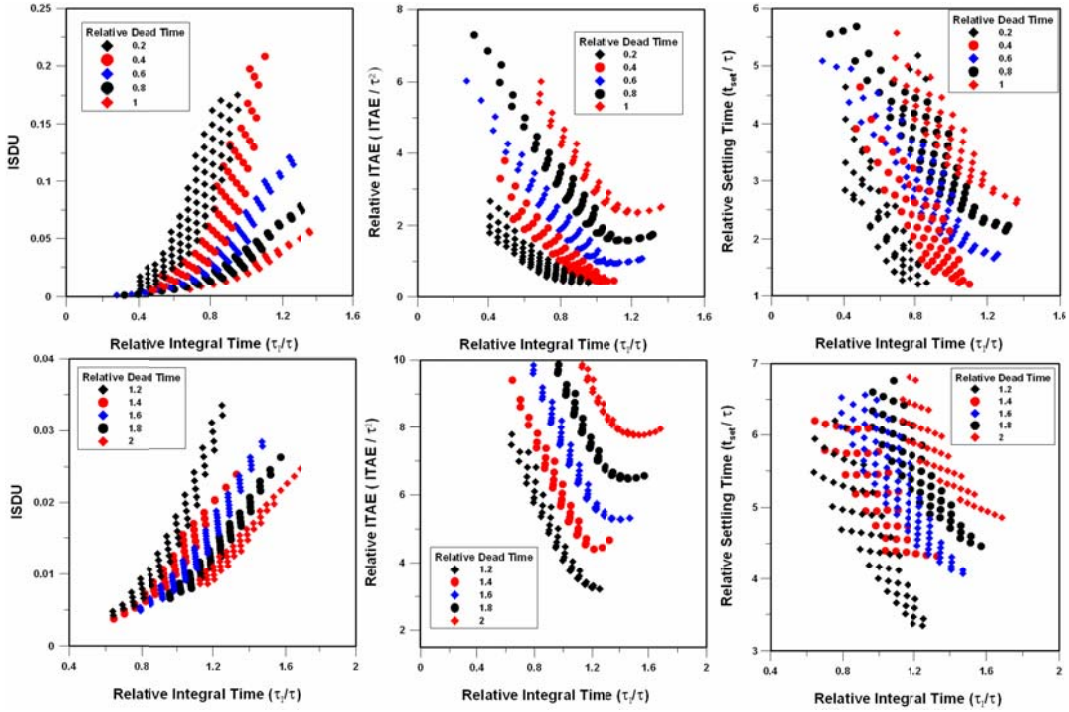

Figure 5: Graph of relative objective functions versus the relative integral time for the generalized controller model for different relative dead times.

The first step in the second tuning method is choosing the desired values of the relative controller performance criteria for a given relative dead time. Using Figure 3, a desired point is first chosen in the output space. This point should be chosen to balance the trade-off between each of the three objectives based on the preferences of the user. From the relative objective function values of the chosen point, the corresponding values of both the relative controller gain and the relative integral time can be determined from Figures 4 and 5. The optimum controller parameters are then calculated for the known process gain and time constant based on the relative input variables. Using this tuning method ensures that controller tuning parameters will be located within the Pareto domain and therefore optimal for the selected performance criteria. The two tuning methods use a different approach to achieve optimum controller performance.

6.0 Application of the Tuning Methods

6.1 First-Order Plus Dead Time System

The developed tuning method was first applied for a specific FOPDT system characterized by a process gain (K_p) of 1.5, a time constant (τ) of 5 and a dead time (θ) of 3. Results obtained were compared to several previously developed PI controller tuning methods.

The simulated FOPDT process corresponds to a relative dead time of 0.6. The Pareto domain for this specific value of the relative dead time was compared to the optimum controller parameters and objectives identified by several previously developed controller correlations. The controller correlations used for comparison along with the objective criteria used in each correlation are shown in Table 2. The results from the comparison are shown in Figure 6.

Table 2: Controller correlations used for comparison, and their objective criteria.

Method	Objective Criteria
Chien and Fruehauf (1990)	IMC
Cohen and Coon (1953)	One quarter decay ratio
Hägglund and Åström (2002)	IAE
Skogestad (2003)	IMC
Smith and Corripio (1997)	ITAE
Tavakoli et al. (2007)	Multi-objective Optimization
Ziegler and Nichols (1942)	One quarter decay ratio

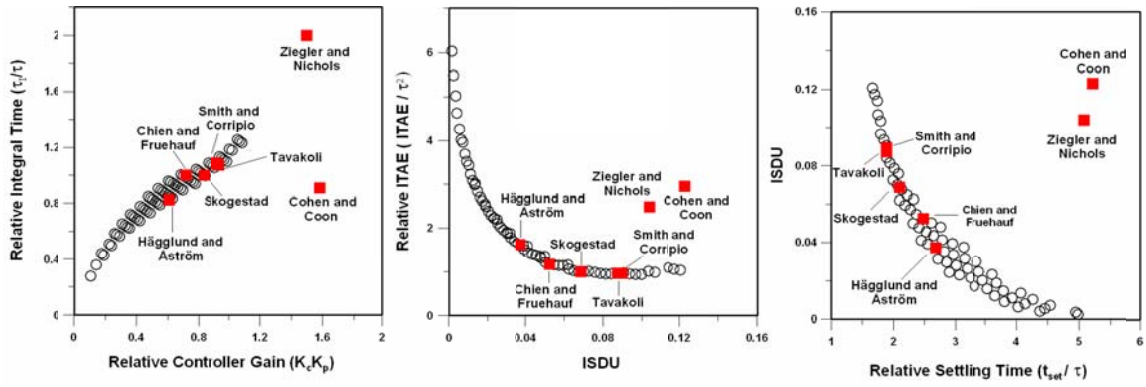

Figure 6: Comparison of the PI controller Pareto domain with other PI controller tuning methods.

Figure 6 clearly demonstrates that the tuning method developed in this investigation provides a general framework for selecting PI controller parameters that would systematically be optimal based on the three performance indicators used to circumscribe the Pareto domain. The choice of the PI controller parameters depends on the performance specifications of control engineers. The PI control parameters of the two earlier control algorithms, Ziegler-Nichols and Cohen-Coon, lie outside the Pareto domain and all three performance criteria are therefore worse than a particular point located within the Pareto domain. On the other hand, the other five tuning methods provide controller parameters that are located on the Pareto domain, as they minimize at least one of the objective criteria used in this investigation.

These results clearly show that the Pareto domain offers an enhancement to previously developed controller correlations, since all possible optimum values for the PI controller parameters can be considered before choosing final controller parameters.

6.2 Fourth Order Plus Dead Time System

To test the proposed tuning method for higher-order systems, it was implemented for the control of an open-loop stable fourth-order system (with all real poles) with dead time. Although the tuning method was developed for a FOPDT system, a fourth-order system can be adequately approximated using a FOPDT model, and the tuning procedure developed in this study can be applied. The fourth-order system used in this study and the approximated FOPDT system are given in Equations (10) and (11), respectively. The FOPDT parameters were obtained by minimizing the squares of the differences between the FOPDT open-loop response and the fourth-order open loop response. The responses of both systems are shown in Figure 7.

$$\frac{y(s)}{u(s)} = \frac{1.5\, e^{-s}}{(3.5s + 1)(2.5s + 1)(1.5s + 1)(0.75s + 1)} \tag{10}$$

$$\frac{y(s)}{u(s)} = \frac{1.51\, e^{-4.29s}}{(5.32s + 1)} \tag{11}$$

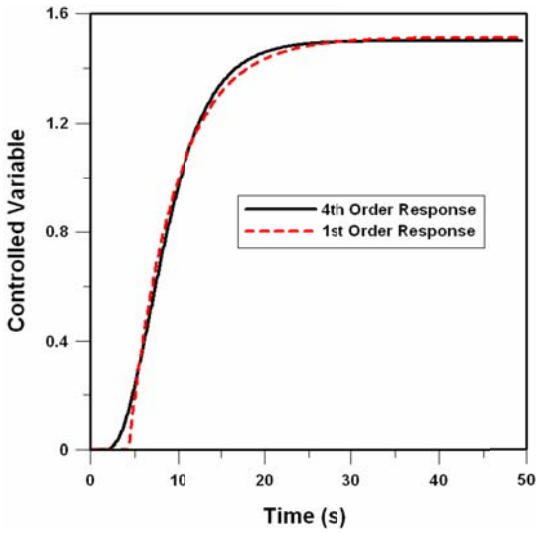

Figure 7: Open loop response for the simulated fourth-order system and FOPDT system.

Figure 7 clearly shows that the FOPDT system fits very closely the fourth order response. The tuning method discussed in Section 5.2 was next performed using the FOPDT process parameters. The relative dead time for this system was equal to 0.8, and the chosen point from Figure 3 was based on a relative ITAE value of 3, which demonstrates a good compromise between each of the three objectives. For a relative ITAE value of 3 and a relative dead time of 0.8, the corresponding values of K_c and τ_I were found to be 0.264 and 4.26 based on Figures 4 and 5. Both the fourth order system and the FOPDT system were controlled using these controller parameters for a unit set point change. The closed-loop response of both systems is shown in Figure 8 and the corresponding objective values are shown in Table 3.

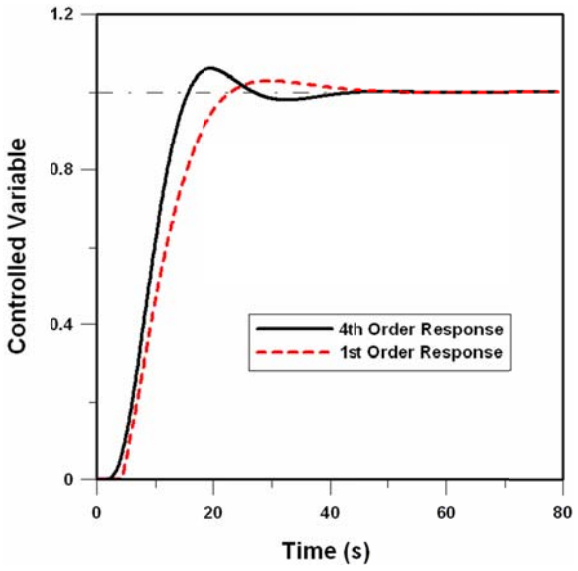

Figure 8: Closed loop response for the simulated fourth-order system and FOPDT system.

Table 3: Performance criteria for the simulated fourth-order and FOPDT systems for unit set point change.

System	ISDU	ITAE	Settling Time
Fourth Order System	0.0018	86.03	22.7
FOPDT System	0.0250	84.91	20.9

Figure 8 and Table 3 clearly demonstrate that the controller configuration determined using the developed tuning method was effective in controlling the fourth order system. Both systems showed very similar responses, with all three performance criteria very close to each other. These results show that the proposed tuning method can be applied to fourth order systems as well as FOPDT systems.

6.3 Application to a Process with a First-Order Disturbance

The proposed tuning method was also evaluated for the case of a disturbance. The controller parameters determined in Section 6.2 were used to control the same fourth order system subject to a unit set point change for a first order disturbance. The first order disturbance transfer function had a gain of 1.5 and a time constant of 3 time units as shown in Table 4. The closed loop response of the fourth order system is shown in Figure 9 and the associated performance criteria are presented in Table 4.

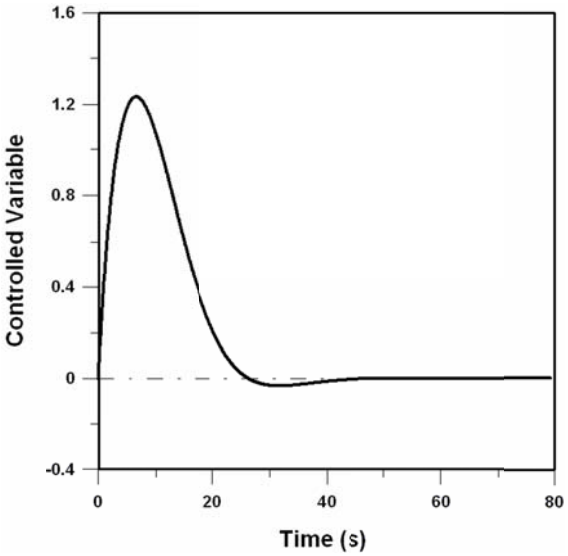

Figure 9: Closed loop response for the simulated fourth-order system subject to a first order disturbance.

Table 4: Parameters of the first order disturbance and the resulting objective functions.

$K_{disturb}$	$\tau_{distrub}$	ISDU	ITAE	Settling Time
1.5	3	0.00091	161.8	24.0

Figure 9 and Table 4 show that the developed tuning method performed well for the fourth order process with a first order disturbance. Similar settling time was realized for this simulation as for both responses shown in Section 6.2. The ITAE was larger for the response to a disturbance, but the ISDU was reduced. All of the results from Section 6 clearly show that the tuning method developed by approximating the Pareto domain leads to high controller performance, and is applicable to a wide variety of processes.

7.0 Conclusions

In this paper, PI controller tuning methods were developed considering multiple objective criteria. The methods were developed by optimizing the ITAE, ISDU, and settling time for a FOPDT system. It was found that a strong correlation exists in the Pareto domain between the two controller input parameters, the relative controller gain and the relative integral time. This implies that when configuring the controller, only one of the controller parameters needs to be specified, as the other is obtained via the strong correlation.

Using the controller optimization results, two methods were proposed for tuning the PI controller. The first tuning method allows for optimum controller performance to be obtained by initially specifying either one of the controller input parameters. The second tuning method involves first specifying the preferred relative objective function values from the Pareto domain, which correspond to specific values of the controller parameters. The developed controller tuning methods were compared to several previously developed controller correlations. It was found that all previously developed controller correlations showed equal or worse performance than that identified by the Pareto domain, but with the limitation of not allowing for enhanced understanding of the many optimum solutions and the trade-off between each performance criterion. Finally the tuning methods were applied to a fourth order process and a process with a disturbance, and were shown to perform well for these two applications.

8.0 Nomenclature

Variable	Definition	Unit
b	Controller input space intercept	dimensionless
f	Objective function in optimization problem	varies
$ISDU$	Integral of the Squares of the Differences in the Manipulated Variable	varies
$ITAE$	Integral of the Time Weighted Absolute Error	varies
K_c	Controller Gain	units of u/y
K_p	Process Gain	units of y/u
m	Controller input space slope	dimensionless
n	Number of input variables	dimensionless
P	Point in the Pareto domain	dimensionless
t_{set}	Settling Time	s
u	Manipulated variable	varies
x	Input function in optimization problem	varies
y	Controlled variable	varies
Greek Symbols		
Δt	Time step	s
ε	Error	varies
θ	Dead time	s
τ	Time constant	s
τ_I	Integral Time	s

Subscript	Definition
t	Time

9.0 References

Chien, I-L., and P.S. Fruehauf, "Consider IMC Tuning to Improve Controller Performance," Chem. Eng. Progress. **86**, 33-41 (1990).

Cohen, G.H., and G.A. Coon, "Theoretical Considerations of Retarded Control," Trans ASME. **86**, 827-834 (1953).

Cvejn, J., "Sub-optimal PID controller settings for FOPDT systems with long dead time," J. Process Control, **19**, 1486-1495 (2009).

Deb, K.A., A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE* Trans. on Evol. Comput. **6**, 182-197 (2002).

Desbourough, L. and R. Miller, "Increasing customer value of industrial control performance monitoring Honeywell's experience", in "Proc. 6th Int. Conference on Chem. Proc. Control, F. Rawlings, Ogunnaike, B. and Eaton, J., Eds., USA, January, 2001, AIChE, USA (2002), pp. 172-192.

Fonseca, C., "Multiobjective Genetic Algorithms with Application to Control Engineering Problems," PhD Thesis, University of Sheffield (1995).

Hägglund, T., "An Industrial Dead-time Compensating PI Controller," Control Eng. Pract. **4**, 749-756 (1996).

Hägglund, T. and K.J. Åström, "Revisiting the Ziegler-Nicholas Tuning Rules for PI Control," Asian J. Control, **4**, 364-380 (2002).

Halsall-Whitney H. and J. Thibault, "Multi-objective optimization for chemical processes and controller design: Approximating and classifying the Pareto domain," Comput. Chem. Eng. **30,** 1155-1168 (2006).

Koo, D.G., J. Lee, D.K. Lee, C. Han, L.S. Gyu, J.H. Jung, M. and Lee, "A Tuning of the Nonlinear PI Controller and Its Experimental Application," Korean J. Chem. Eng. **18**, 451-455 (2001).

Kookos, I.K., “PI Controller Tuning via Multi-Objective Optimization”, in “Proceedings 7th Mediterranean Conference on Control and Autom.,” Z. Palmor, Ed., Israel, June 28-30, 1999, MCA, USA (1999), pp. 408-419.

Lee, S. and J.H. Park, “Performance improvement of PI controller with nonlinear error shaping function: IDA-PBC approach,” Appl. Math. Comput. **215,** 3620-3630 (2010).

Logist, F., P.M. Van Erdeghem, and J.F. Van Impe, “Efficient deterministic multiple objective optimal control of (bio)chemical processes,” Chem. Eng. Sci. **64**, 2527-2538 (2009).

Madhuranthakam, C.R., A. Elkamel, and H. Budman, “Optimal tuning of PID controllers for FOPTD, SOPTD and SOPTD with lead processes,” Chem. Eng. and Process. **47**, 251-264 (2008).

Marlin, T., “Process Control: Designing Processes and Control Systems for Dynamic Performance,” John Wiley & Sons, Inc., USA (2001).

Roy, A. and K. Iqbal, “PID controller tuning for the first-order-plus-dead-time process model via Hermite-Biehler theorem,” ISA Trans. **44**, 363-378 (2005).

Seborg, D.E., T.F. Edgar, and D. Mellichamp, “Process Dynamics and Control,” John Wiley & Sons, Inc., USA (2004).

Skogestad, S., “Simple Analytic Rules for Model Reduction and PID Controller Tuning,” J. Process Control. **13**, 291-309 (2003).

Smith, C.A. and A.B. Corripio, “Principles of Automatic Control,” John Wiley & Sons, Inc., USA (1997).

Stephanopoulos, G., “Chemical Process Control: An Introduction to Theory and Practice,” Prentice-Hall, Inc., USA (1984).

Tan, K.K., T.H. Lee and F.M. Leu, “Predictive PI versus Smith control for dead-time compensation,” ISA Trans. **40**, 17-29 (2001).

Tavakoli, S., I. Griffin and P.J. Fleming, “Multi-Objective Optimization Approach to the PI Tuning Problem,” in “Proc. of the 2007 IEEE Congress on Evol. Comp.,” J. Xu, Ed., Singapore, September 25-27, 2007, IEEE, USA (2007), pp. 3165-3171.

Vandervoort A., “Multi-objective optimization techniques and their application to complex engineering problems,” M.A. Sc. Thesis, University of Ottawa (2010).

Xue, Y., and L. Donghai and F. Gao, “Multi-objective optimization and selection for the PI control of ALSTOM gasifier problem,” Control Eng. Prac. **18**, 67-76 (2010).

Zeigler, J.G., and N.B. Nichols, “Optimum Settings for Automatic Controllers,” Trans ASME. **64**, 759-768 (1942).

Conclusions and Recommendations

1.0 Conclusions

The research program undertaken in this thesis focused on the development of new multi-objective optimization techniques and their application to complex chemical engineering problems. In Chapters 1 and 3, new multi-objective optimization techniques were developed with the goals of improving the accuracy and efficiency of the Pareto domain approximation relative to current MOO techniques. In Chapters 2 and 4, the new MOO methods were applied to complex chemical engineering problems, specifically to a reactor producing ethylene oxide from ethylene and a PI controller model. A summary of the specific conclusions from each chapter are given below.

In Paper 1, An Objective Based Gradient Algorithm for Approximating the Pareto Domain, the performance of the first method developed for approximating the Pareto domain, the OBGA, was compared to the DPEA and NSGA-II MOO techniques. For the four optimization problems considered in this chapter, the OBGA systematically produced a more accurate Pareto domain than DPEA and NSGA-II. For relatively simple problems, the OBGA led to an increase in efficiency relative to DPEA and NSGA-II. For more complex objective functions, the efficiency of the OBGA was reduced relative to current methods due to an increase in the required number of objective function calls. It was found that even if for a given problem the computation time could be reduced by using genetic algorithms, the OBGA still provides the best approximation of the Pareto domain, and the greatest understanding of the final accuracy obtained.

In Paper 2, Multi-Objective Optimization of an Industrial Ethylene Oxide Reactor, the OBGA was applied to an industrial reactor producing ethylene oxide from ethylene. The approximated Pareto domain was ranked using the Net-Flow method. Based on the ranked Pareto domain it was recommended that the ethylene-oxide reactor be operated at high values of the inlet pressure and gas temperature, and low values of the inlet volumetric gas flowrate and chemical reaction moderator concentration, corresponding to the best compromise or trade-off solution between each of the three objectives. It was also recommended that the inlet pressure and volumetric gas flowrate variables be controlled carefully, as small variation in the gas flowrate, or a decrease in the inlet pressure could lead to a solution outside of the Pareto domain, with deteriorated performance.

In Paper 3, A Principle Component Grid Algorithm for Approximating the Pareto Domain, the performance of the second developed method for approximating the Pareto domain, the PCGA, was compared to the GSA, and NSGA-II MOO techniques. For all problems tested it was found that the PCGA produced a more accurate Pareto domain than both the GSA and NSGA-II, and spanned the Pareto domain more evenly and completely. It was also found that the PCGA approximated the Pareto domain for each optimization problem with the highest efficiency of all three algorithms.

In Paper 4, New PI Controller Tuning Methods using Multi-Objective Optimization, the PCGA was applied to the development of new PI controller tuning methods. Two tuning methods were developed based on the Pareto domain to achieve optimal controller performance by specifying either one of the controller input parameters or the desired values of the performance criteria. The developed controller tuning methods were compared to several previously developed controller correlations. It was found that all previously developed controller correlations showed equal or worse performance than that based on the Pareto domain. In addition, previous correlations provide limited insight with respect to the trade-off between each performance criterion. Finally, the tuning methods were applied to a fourth order process and a process with a disturbance, and demonstrated excellent performance.

2.0 Recommendations

The research program undertaken in this thesis involved the development of two new methods for approximating the Pareto domain. For many optimization problems, after approximating the Pareto domain the various solutions must be ranked to aid in the selection of final operating conditions. In this research program Net-Flow was used to rank various solutions within the Pareto domain. There are also alternative methods, such as the Rough Set method, for ranking solutions within the Pareto domain. If further research were performed in the area of multi-objective optimization it would be recommended to study these various raking methods. The various methods could be compared and evaluated in terms of the ease of use, robustness, and ability to capture the decision maker's preferences.

Research into the various methods could also reveal areas of possible improvement, leading to the development of new methods for ranking the Pareto domain.

A second possible area of focus for future multi-objective optimization research relates to the spacing of various solutions within the Pareto domain. Many algorithms for approximating the Pareto domain have been developed with the intention of producing a well distributed Pareto domain. For example, NSGA-II considers the distance between various solutions when approximating the Pareto domain, but at the cost of increased complexity relative to more simple methods of approximating the Pareto such as DPEA. It is therefore recommended that the importance of producing a well distributed Pareto domain is studied. For example, it should be determined if the spacing of the Pareto domain has a significant effect on the ranking scores obtained for each solution using various ranking methods. If it is determined that the highest ranked solutions are consistent regardless of the spacing found in the Pareto domain, than producing a well distributed Pareto domain may not prove worthwhile due to the corresponding increase in complexity of the Pareto domain approximation. On the other hand, if it is determined that the ranking scores change significantly with changing spacing of the solutions within the Pareto domain, choosing the Pareto domain approximation with the most evenly distributed solutions could prove beneficial.

Finally, further research could be performed on the implementation of the Objective-Based Gradient Algorithm and the Principal Component Grid Algorithm. In this research, preferred values of several parameters used in the PCGA and the OBGA were determined for the problems studied. If further research were performed in this area, optimum algorithm parameters could be determined for a wider range of problems, leading to the development of general rules for the implementation of the OBGA and PCGA. General recommendations could also be made in relation to the types of optimization problems that are ideally suited for the OBGA and PCGA, and the types of problems for which alternative MOO techniques may be preferable.