LEAPFROGGING WITH IMPROVED INITIALIZATION AND ITS APPLICATIONS: DEMONSTRATION OF LEAPFROGGING ON HORIZON PREDICTIVE CONTROL OF A HEAT EXCHANGER

By

UPASANA MANIMEGALAI SRIDHAR

Bachelor of Technology in Chemical Engineering
Sri Sivasubramania Nadar College of Engineering, Anna University
Kalavakkam, Tamil Nadu, India
2009

Master of Science in Chemical Engineering
Oklahoma State University
Stillwater, Oklahoma, USA
2010

Submitted to the Faculty of the
Graduate College of the
Oklahoma State University
in partial fulfillment of
the requirements for
the Degree of
DOCTOR OF PHILOSOPHY
July, 2014

LEAPFROGGING WITH IMPROVED INITIALIZATION AND ITS APPLICATIONS: DEMONSTRATION OF LEAPFROGGING ON HORIZON PREDICTIVE CONTROL OF A HEAT EXCHANGER

Dissertation Approved:

This dissertation is dedicated to my grandmother Suguna who played a major and a very important role in my growth. This dissertation is also dedicated to my mother Manimegalai who wanted to be with me for more than a year. This work is one of their dreams. Both of them, if alive, would have felt happy to know about their fulfilled dream.

ACKNOWLEDGEMENTS

First and foremost I would like to express my deepest gratitude to my advisor Dr. R. Russell Rhinehart. I am grateful for his relentless guidance, insurmountable patience, unconditional support, and encouragement. Completing the research work in the best possible manner wouldn't have been possible without his immeasurable guidance and inputs through the years. Every interaction with him played a role in developing myself both professionally and personally. It is a real honor to be a part of Dr. R. Russell Rhinehart's research team. I also thank him for providing me with financial assistance during my period of stay at OSU.

I would like to express my heartiest thanks to my committee members Dr. Kenneth J. Bell and Dr. James R. Whiteley for serving on my committee and providing constructive criticism for improvement. Thanks to Dr. Arland H. Johannes for serving on my committee and also believing in me for a role as ENGR 1412 teaching associate. I am indebted to Dr. Anand N. Vennavelli for his interest in my career development by providing considerable comments for both personal and professional development and also serving on my committee.

With deep sense of gratitude I would like to thank my brother and lab mate Anand Govindarajan who has been a constant source of support and encouragement throughout. I am greatly indebted to his interest in my research and providing inputs during every phase of my research work. He has been an unfailing source of support for the past 7 years.

I would like to thank and pay gratitude to each and every one involved with OSU Automation Society. Being a part of this society has created a platform to network and grow professionally.

Special mention and thanks to Mr. Mike R. Resetarits and Dr. Maurice J. Wilkins for their interest in my career development.

A special thanks to Dr. Karen A. High for her support throughout with co-instructing ENGR 1412. I owe my sincere thanks to Dr. Sundararajan V. Madihally and the School of Chemical Engineering for having belief in me and providing financial assistance throughout my period of stay at OSU. Thanks to ChE staff Eileen Nelson, Shelley Taylor, Shelley Potter, Paula Kendrick and Carolyn Sanders for their timely administrative help with my journey at the School of Chemical Engineering.

I am very much indebted beyond ambit of terms to my lab mates Dr. Anirudh Patrachari for his constant guidance, support, help with ENGR 1412 and for being a continuous source of inspiration and Dr. Solomon Gebreyohannes for his continuous encouragement and support. Thanks to Rama Rao, Lalitha and their families for creating a homely atmosphere for me and my friends periodically during out stay away from home.

I thank Kumar for lending a hand of support during all difficult times. Thanks to Jeet for being a continuous pillar of support. I would like to thank all my friends Venky, Suresh, Viji, Sunil, Mounika, Krishna, Shubha, Preethi, Kinnera, Vidvath, Samyukta, Harita and Yuvraj for their constant encouragement and interest in my wellbeing. I owe my thanks to Akshaya, Revathi and Pallavi for their continuous support during my stay for an internship at Houston. Special thanks to Govindarajan and Jayashree for their continuous advice and support throughout my studies at OSU. Thanks to Lakshmi and Nageswara Rao Lakkakula for being a huge support during difficult times.

I would be failing in my duty if I don’t thank my all my undergraduate professors at Sri Sivasubramania Nadar College of Engineering, India, who have been instrumental in developing the fundamentals of Chemical Engineering. Special thanks to Dr. S. Rajasekaran who played an important role in pursuing a Chemical Engineering undergraduate degree.

Thanks to my father Sridhar for what I am today. I am pretty sure I wouldn’t have done PhD but for his persistent encouragement during every other phone call during my Masters on pursuing a PhD. I owe a major portion of my PhD and my growth both professionally and personally to him. Thanks to my mother Kalavathi for the pains she took to help me get into Chemical Engineering. That was the starting point for everything I am today.

Heartiest thanks to Subba Rao and Bharani, my second parents for their unconditional love and support during any phase of my life. Thanks to my cousin Rajes for his financial support during my stay at USA.

I would like to acknowledge my entire family and friends for being a moving spirit and helping me in all my pursuits. Thanks to all the kids who stayed at our house and grew with me right from my school days. Special mention to soon to be Dr. Govindaraj and Dr. Prasath with whom I share a special bond. Every stage of their lives has been a constant source of inspiration for me. Thanks to every member of Rural Education Welfare and Resource Development (REWARD) for giving me an opportunity to be a part of the REWARD family. Every interaction with REWARD teachers and students has truly been inspiring and a learning experience.

Name: UPASANA MANIMEGALAI SRIDHAR

Date of Degree: JULY 2014

Title of Study: LEAPFROGGING WITH IMPROVED INITIALIZATION AND ITS APPLICATIONS: DEMONSTRATION OF LEAPFROGGING ON HORIZON PREDICTIVE CONTROL OF A HEAT EXCHANGER

Major Field: CHEMICAL ENGINEERING

Abstract:

Leapfrogging (LF) is a recently developed optimization technique that initially places players in random spots in the feasible decision variable space [1]. The approach to reach the global optimum is by “Leaping” the player with the worst objective function (OF) value “Over” the player with the best OF value into the reflected hyper-volume that connects the player with the best and the worst OF until the players converge at an optimum [1]. LF has several advantages compared to other optimization techniques in terms of computational efficiency and higher probability to reach the global optimum [1]. This is demonstrated in several applications [1-7].

The main focus of this work is to develop LF [8] by exploring the initialization step through a fundamental analysis and supporting the developed technique with mathematical truths. In this improvisation the OF surface is initially explored with a large number of players, the players are sorted in ascending order of their OF values and the top few players are selected to continue with the optimization technique. This improvement is found to increase the probability of finding the global optimum as one of the initial players is placed in the vicinity of the global optimum during initialization and thus draws all the other players towards it.

In order to establish the applicability of LF and its improvement on process engineering applications, this work focusses on implementation of original and modified LF to model a pilot scale Shell and Tube Heat Exchanger (HX 001) process, which is nonlinear. Steam is used to increase the temperature of the water on the cold side. For this study, the outlet temperature of the cold side fluid is considered as the control variable (CV). The hot side steam valve opening is considered as the manipulated variable (MV). This CV-MV relation is modeled using original and modified LF to find the model parameters that best fit the experimental skyline function generated for modeling purpose in the Unit Operations Lab, OSU-Stillwater. Next, the model parameters are used to implement a horizon predictive control on the HX001 process to control the CV.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure Page

CHAPTER 1

INTRODUCTION

1.1 Optimization:

Optimization stands out as the fundamental step in chemical and process engineering applications such as in process modeling, advanced process control, real time optimization, measurement, scheduling, process design, product design, fault detection analysis, data reconciliation, etc.[9]. Optimization applications in such areas, express challenges because of the presence of multi-optima, surface aberrations, constraints, multivariable impacts, multi-objective, nonlinear surfaces, stochastic penalties, discontinuities, etc. [10-18]. Hence, the chemical and process engineering community explored the applicability of various different optimization techniques classified as gradient based, evolutionary, direct search, dynamic programming, differential evolution, etc. [10-18] on these problems.

An efficient optimization technique is required to handle such optimization problems [10]. The desirable qualities of an efficient optimization include [19, 20]:

1. Global Optimum:

   Optimization problems might express many optima which include both local and global optimum. Finding the global optimum amongst the presence of multiple optima is a desirable quality of an efficient optimization technique.

2. Accuracy:

   While the desirable property of an efficient optimization technique is to find the global optimum amidst the local optima, the efficient optimization technique should be accurate enough to be able to converge either at the global optimum or within the vicinity of the global optimum with a higher probability.

3. Robustness and Flexibility:

   Optimization problems might express surface aberrations in terms of nonlinearity, discontinuities etc. An efficient optimization technique should be able to handle such surface aberrations. It should be able to handle any type of constraints and topographical difficulties.

4. Less Computational Burden And More Speed:

   An efficient optimization technique should be able to find the global optimum with less computational burden without compromising on the quality of the solution relative to an accepted standard. Optimization techniques require computing the objective function (OF) value and each computation of OF function is regarded as a function evaluation. The lower the number of function evaluations, higher is the speed of the optimization technique. An efficient optimization technique should be able to converge at the global optimum with less number of function evaluations.

5. User-friendliness And Simplicity:

   The algorithm of an efficient optimization technique should be user-friendly, for anyone to understand and implement the algorithm for any optimization problem. An efficient optimization technique should have a procedure that is easily understandable. At the same time it should have fewer lines of code and less number of user-defined inputs for executing the technique for any user to be able to implement it for any application.

6. Scalability:

   An efficient optimization technique should be scalable to any number of dimensions.

Leapfrogging is a recently developed novel meta-heuristic, optimization technique that initially places players in random spots in the feasible decision variable (DV) space [21]. The approach to reach the global optimum is by “Leaping” the player with worst OF value “Over” the player with the best OF value into the reflected hyper-volume that connects the player with the best and the worst OF value [21]. The optimum is said to be attained when all the players converge in a common DV “Spot” [21]. Leapfrogging has several advantages compared to other optimization techniques in terms of computational efficiency and higher probability to reach the global optimum [21]. This was demonstrated on several two-dimensional functions and several high dimensional process systems engineering applications [1-4, 7].

However, a fundamental analysis supporting the Leapfrogging optimization technique with mathematical truths has not been executed, yet. Providing a fundamental knowledge of the optimization technique is required to finely understand the technique and provide scope for further improvement of the technique.

1.2 Horizon Predictive Control Optimization Application [22], [23-25]:

Industries widely use optimization techniques for optimizing an economic objective in real time. The optimizers that are used for this purpose are called real time optimizers (RTO). These RTO then set the set point targets for intelligent controllers such as horizon predictive controllers (HPC). HPCs have proved themselves to be beneficial controllers [23-25].

The HPCs use an optimization technique within them. The optimization technique running within the HPC based on the set point target optimizes the controller action in order to make the process reach the set point. Traditionally, the optimization problem of the HPC is linearized and linear programming is used to find the best set of control action to make the process reach the set point values [26, 27].

In this study Leapfrogging will be tested for its applicability for a HPC application. Since Leapfrogging is a newly developed technique, testing its applicability on HPC will showcase the adaptability of Leapfrogging for an advanced process control application. This application will demonstrate advance understanding of Leapfrogging optimization and its utility in a process control environment. This application will also reveal proof-of-concept in Leapfrogging as applicable to application community.

HPC for a 4 pass Shell and Tube Heat Exchanger (HX) unit will be established using LabView and Leapfrogging optimization technique will be used to find the best set of control actions to control the tube side fluid outlet temperature.

1.3 Hypothesis:

The primary aim of this project is to perform a fundamental analysis of the Leapfrogging optimization technique which would pave way towards improvement. A secondary aim is to establish credibility of the optimizer to the process control community or industry by testing the technique and its improvements on a process systems engineering application.

Improving the optimization technique by providing a fundamental analysis will lead to new knowledge. The successful application of the original and improved Leapfrogging technique on a pilot scale shell and tube HX unit and implementation of nonlinear HPC on the process to control the outlet temperature of the cold side fluid will establish credibility.

1.4 Specific Aims:

Based on the hypothesis, the research plan is divided to into three specific aims. The specific aims of this work are to:

1. Provide fundamental knowledge and improve effectiveness of the Leapfrogging technique:

The main focus of this work is to improve the Leapfrogging optimization technique by providing fundamental knowledge about the technique, supporting the heuristically (experience based) developed technique with mathematical truths and analyzing the initialization of the technique. This project explores the requirement and effect of an initial surface exploration of the OF surface. Through the initial surface exploration analysis, the plan is to increase the probability of finding the global optimum which may increase the efficiency of the technique. Efficiency is a measure of finding the global with minimum number of function evaluations.

2. Apply Leapfrogging and Leapfrogging with improved initialization on nonlinear pilot scale Heat Exchanger (HX) modeling:

In order to establish the applicability of Leapfrogging and its improvement on process systems engineering applications, both original and improved Leapfrogging are used to regress a pilot scale shell and tube HX process, which is nonlinear. Experimental data using a skyline function is generated in the Unit Operations Lab, Oklahoma State University (OSU), Stillwater. Original and improved Leapfrogging are used to find the best set of model parameters.

3. Implement constrained nonlinear HPC on the HX and use Leapfrogging to solve for future manipulated variable (MV) moves:

In order to establish credibility of the optimization technique, Leapfrogging is used for demonstrating its applicability to nonlinear horizon predictive control on the pilot scale HX equipment. Leapfrogging is used for computing the future moves of the manipulated variable at every time instant, in order to make the controlled variable, reach the desired set point, along a desired path while avoiding constraints.

1.5 Broad Impact:

The fundamental analysis of this technique along with supporting mathematical truths will provide a platform to build and develop the Leapfrogging optimization technique which paves way to new knowledge. Applications of original and improved Leapfrogging optimization technique on HX modeling and establishing nonlinear HPC of a HX will demonstrate implementation, credibility and understanding of the novel optimization technique. This application demonstration will explore the possibility of using Leapfrogging optimization technique in the chemical industries.

1.6 PhD Contributions:

The contributions to real world based on this research work are as follows:

1. Improved Initialization:

   This work will present the idea of improved initialization on Leapfrogging optimization. A mathematical analysis of the improved initialization idea will also be presented.

   Experimental support will be provided in order to reveal credibility in concept and analysis. Simulated benefit to society in grounding and proof of concept analysis will open doors for others to translate the idea in other areas. This contribution is aimed more at providing fundamental analysis of the Leapfrogging technique and guide for well-known problems.

2. Application Credibility for Practitioners:

   This work will demonstrate the applicability of Leapfrogging and Leapfrogging with improved initialization on HX modeling and Leapfrogging for HPC implementation on HX. Both these applications exhibit surface aberrations by possessing nonlinearity and constraints. The HPC application will be demonstrated on a real HX process.

This will reveal practicality by showcasing the robustness, computational speed and constraint handling nature of the optimization technique.

3. Leapfrogging Development and Exploration:

   This research work is built on the Leapfrogging optimization technique. The development and exploration of this optimization technique will be explained in the background section. Significant contributions were made in developing and exploring the Leapfrogging optimization technique in from its conception in 2010.

4. Accelerated Convergence Studies

   In order to accelerate the convergence of the original Leapfrogging optimization technique, studies were conducted on the leap-to window size. Original Leapfrogging uses the size of the hyper-volume connecting the best and the worst player for the size of the reflected hyper-volume into which the worst players leaps over. Significant contributions are made in the

   a. Accelerated convergence studies of Leapfrogging optimization
   b. Demonstration of Leapfrogging and its improvement on the steady state modeling of a 11 equilibrium staged distillation column
   c. Implementation of HPC on a pilot scale 5 trayed distillation column in the Unit Operations Lab, OSU-Stillwater.

This dissertation will focus on the improved initialization idea and its application credibility for practitioners. The Chapter 2 (Background) section will explain the development and exploration of the Leapfrogging optimization technique.

CHAPTER 2

BACKGROUND

2.1 Conventional Optimization Algorithm:

Optimization is being widely used in chemical and process engineering for various applications discussed in Chapter 1 (Introduction). Optimization applications are broadly classified into continuous and discrete variable cases [9]. Discrete variable optimization cases are those situations in which the DVs are restricted to discrete or finite set of values, for example: the pipe diameter required to transport petroleum from one place to another in the best possible manner without much wastage. Integer or discretized values for the DV is required at the end of implementation of an optimization procedure. On the other hand, continuous variable optimization cases are those in which the DVs can assume real values, for example: finding the optima in Golstein-Price function. The optimization problems are solved iteratively where each iteration uses the information from the previous iteration. The iterations progressively converge to a solution.

Combinatorial approaches are used to solve discrete variable optimization cases [28]. A continuous approach is also used to solve discrete variable optimization case where the problem is reformulated to be considered within a continuous space [28]. Continuous variable optimization techniques can also be used to solve discrete variable optimization cases wherein the final optima is rounded to the closest discrete value of the variables.

Continuous variable optimization cases are solved using two major classes of optimization techniques. One class of optimization techniques are called Linear Programming (LP) techniques where the process and the constraints are linearized before implementing LP on them.

LP was developed for use during World War II in the 1939 [29]. The solution of this optimization approach is always at the constraint intersections. LP has proved its ability to solve problems with millions of DVs and constraints [9, 30]. Though this technique finds the solution with less computational burden, it does not guarantee finding the global optimum because the possible solution can only be at the constraint intersections, not in the interior space. Another major disadvantage of LP is linearizing the process and constraints which means the solution is an approximation.

On the other hand, the other class of optimization techniques to solve nonlinear objective function problems is called nonlinear programming (NLP) technique. The surface demonstrated by a nonlinear objective function surface can either be convex or nonconvex. Multiple local optima may be seen in a nonconvex surface [9].

The NLP techniques are further classified into gradient based techniques and direct search based techniques. Gradient based techniques involve computation of gradient or Hessian or both. Direct search techniques do not require computation of gradient or Hessian elements.

Successive quadratic programming is one of the oldest NLP techniques. The solution can be in the interior region, but substantial computational work is required for iterations. This technique may end up finding the local optimum. An optimization problem with surface aberrations can misdirect the search. This algorithm can be used for solving an extensive range of process engineering problems [9, 31, 32]. This technique linearizes the constraints and assumes the OF surface as a quadratic model. This technique boils down to Newton’s method in an unconstrained situation [33-35].

Levenberg-Marquardt (LM) technique [36], which is also a gradient based NLP technique, is used in locating the optimum of nonlinear functions. It is a blend of the steepest descent and Newton’s method. This technique follows the incremental steepest descent algorithm when away from the solution and switches to Newton’s method when closer to the optimum which would find the solution quickly.

However, since the switch condition is pre-set, it may not be optimal for a particular application. Successive quadratic and other gradient-based optimization techniques such as LM have the disadvantage of encountering zero-valued gradient and zero-valued Hessian elements on their path to the optimum.

The gradient based optimization techniques cannot be used if the gradient information is not trustworthy or unavailable because the optimum is said to be attained only if the gradient value vanishes toward zero [37]. Newton’s second derivative and successive quadratic methods also seek minimum, maximum, or saddle points and are misdirected by inflection points.

Direct search techniques are a broad class of optimization techniques that need little *a priori* knowledge about the behavior of the OF surface of the optimization problem, and do not need calculation of derivatives [9]. Direct search optimization techniques are efficient, at times the only option for a variety of optimization problems, and can provide guaranteed convergence [37]. However, direct search techniques face difficulty in handling constraints [9, 37].

Meta-heuristic optimizers are another set of optimizers classified under the direct search optimization techniques. Meta-heuristic is a Greek word; meta meaning high level and heuristic meaning experience based. Most of the meta-heuristic techniques are computationally intensive and are player based [38]. These optimization techniques aim at exploring the OF surface to find the solution for the optimization problem. The meta-heuristic optimization techniques are suitable for complicated process engineering applications [2-4, 7, 39] because they have less difficulty than techniques that employ gradient and hessian values. Because of their wide range of applications significant amount of research is conducted on studying various different meta-heuristic optimization techniques.

Differential Evolution, Particle Swarm and Scatter Search are few of the meta-heuristic techniques based on population [10, 18, 40, 41]. The population represents placement of players in the DV space. A player is the DV point at which the OF is calculated during each iteration of the optimization technique. The major advantage of these techniques is that, they do not need a good initial guess unlike gradient based techniques because they start with random initial placement of players which end up searching a wide region of the OF surface. A disadvantage of these approaches is that careful evaluation of the search criteria should be employed to preserve the required players and the population diversity (population diversity demonstrates the amount of surface exploration being done by the number of players present at that point in time) [42-44]. If the population is not diverse, these techniques encounter premature convergence or get locked at the local optimum instead of finding the global optimum [42-44].

Hence, a direct search technique with an ability to handle constraints, and which possess the required attributes of an efficient optimization technique is required to solve chemical and process engineering applications such as in process modeling, advanced process control, real time optimization, measurement, scheduling, process design, product design, fault detection analysis, data reconciliation, etc.[9].

2.2 Leapfrogging [1]:

Leapfrogging is a recently developed optimization technique based on a set of heuristic rules. Leapfrogging starts by placing players (points at which the OF is calculated) in random locations on the feasible decision variable (DV) surface. The OF values of all the players are calculated, and the player with the worst (highest) OF value and the player with the best (lowest) OF value are found. During each iteration, the worst player (highest OF value) is leapt (relocated) over the best player (lowest OF value) into the reflected hyper-volume connecting the best and the worst player. Each OF value calculation accounts for one function evaluation.

Figure 2-1 pictorially represents a leap over of the Leapfrogging optimization technique in two dimensions. The contour in Figure 2-1 is a representation of the simple ellipse function with global optimum at the center of the innermost contour (shown in the figure as a cross). The leap-to position of the worst player into the reflected hyper volume is given by the Equation (2-1) :

$$x(i)_{\text{leap-to position}} = x(i)_{\text{current best}} - \alpha r_i \left(x(i)_{\text{current worst}} - x(i)_{\text{current best}}\right) \tag{2-1}$$

In Equation (2-1), i represents the dimension of the overall DV space, x(i) represents the i^{th} DV value, current best and worst represent current best and worst players, α represents the reflected window size (generally considered as 1 by balancing benefit to effort) and r_i represents a uniformly distributed random number between [0,1] which determines the location of the leapt over player within the reflected hyper volume. The random number r_i is independent for each DV.

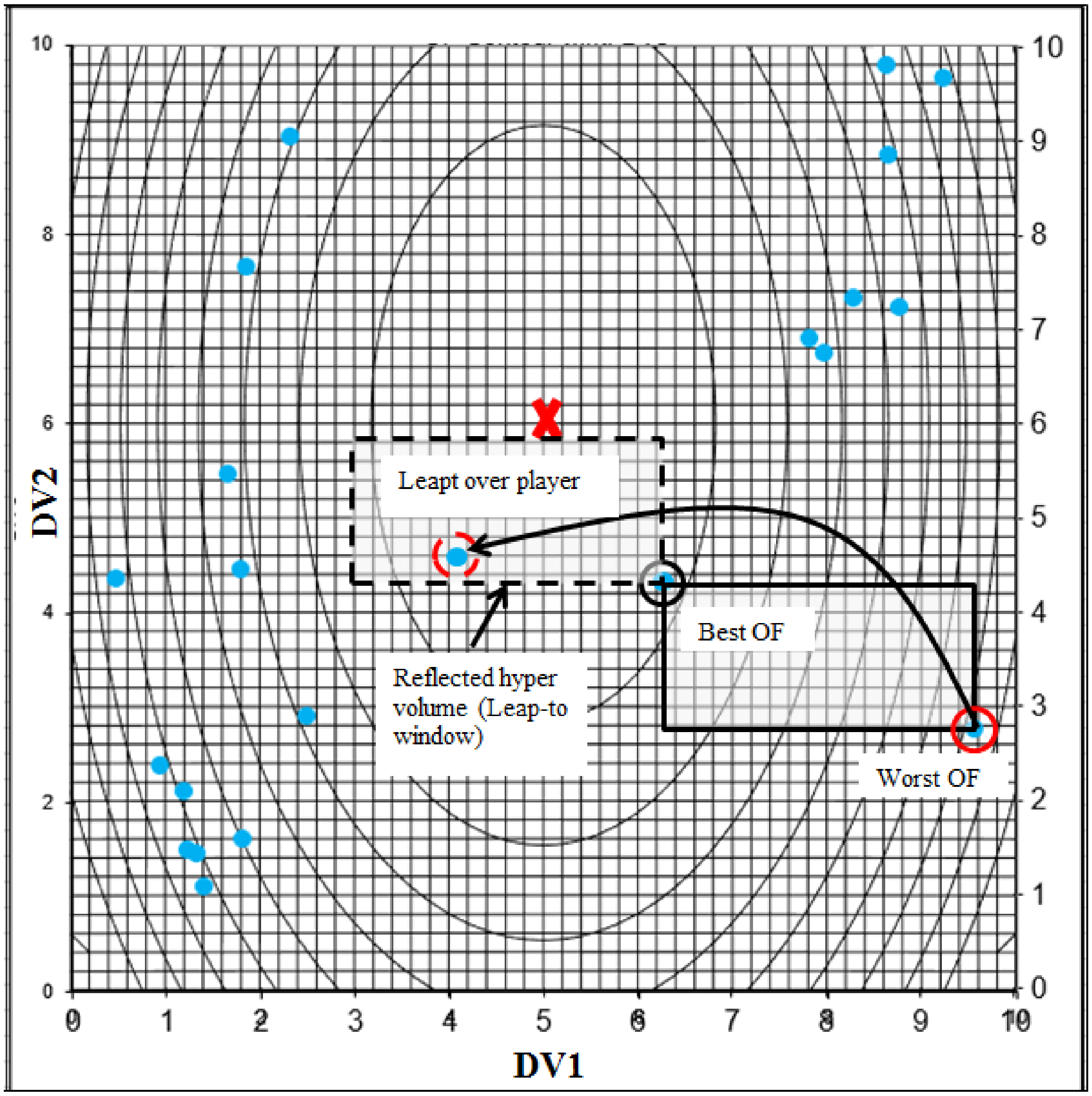


Figure 2-1: Pictorial Representation of a Leap Over

After every leap over, the OF value is evaluated at the new location of the worst player. If the best player has an OF value lesser than the leapt over player, the best continues to be the best. If the leapt over player has a lesser OF value than the current best player, then the leapt over player is reassigned as the best player. Again for the next iteration, the worst player leaps over the best player to a new location. However, OF value computation of all the players is not required because only one player is relocated. The best player will either continue to be the best or the leapt over player's new location will be the new best. The worst player is either worse than or better than the leapt over player. So there is no need to search for the best player again. The leap overs continue until all the players converge within some range of the best. The best of converged OF values is the solution of the optimization problem. This method is just one of many convergence criteria. Different convergence criteria can be used.

In the situation where the DV is discrete and not continuous the leapt over location is rounded to the closest discrete value. Therefore, the Leapfrogging technique can handle both discrete and continuous DV's.

In the situation that the worst player leaps to an infeasible location (constrained region), it leaps from that location back over the current best until it finds a feasible spot. Each time it leaps over to a new spot the OF is evaluated. This shows that Leapfrogging can handle hard constraints or error trapping of any sort.

Many of the direct search optimization techniques proceed by moving the best player to a better spot. In contrast, Leapfrogging technique proceeds by relocating the worst player to a new spot, which is similar to the Nelder Mead simplex approach [45, 46]. But unlike the simplex algorithm which characterizes the locally placed players around the DV-space, Leapfrogging allows any number of players to be placed randomly in the feasible region which helps in a quicker identification of the global optimum.

The multi-player evaluation technique of Leapfrogging is similar to Particle Swarm [40]. But, in Leapfrogging the search distance is cut into half on average during each leap over as opposed to local search exploration of all particles during each iteration of the Particle Swarm technique. This leads to lower number of function evaluations for the Leapfrogging optimization technique as compared to the number of function evaluations of the Particle Swarm technique. The idea behind Particle Swarm, which is also the case in Leapfrogging, is that if a player is in the vicinity (area close by) of the global optimum, it would attract all the players towards it. Leapfrogging is similar to Differential Evolution [10, 18, 41] in terms of employing multi-players and the leap over movement. Differential Evolution proceeds by the logic of survival of the fittest. New players during each epoch (iteration) are generated by adding the difference between randomly chosen vectors. Unlike Leapfrogging, in which always the worst leaps over the best, in Differential Evolution the player selection is random. This provides a specific direction for the Leapfrogging optimization technique towards the global optimum.

Figure 2-2 is a flowchart describing the Leapfrogging optimization technique. The steps are as follows:

1. Randomly evaluate the OF value at random spots in the feasible DV space. These spots are called the players.
2. Find the best player – player with the least OF value.
3. Find the worst player – player with the highest OF value.
4. Leap the worst player over the best into the reflected hyper volume.
5. If the new spot is better than previous best player, then reassign the best player. Otherwise the previous best will continue as the best player.
6. Check convergence criteria.
7. Repeat step 3-6 until all the players converge.

This Leapfrogging optimization technique has to be studied further to provide fundamental knowledge. Underpinning Leapfrogging with theoretical analysis will both establish credibility and lead to improvement of the optimization technique.

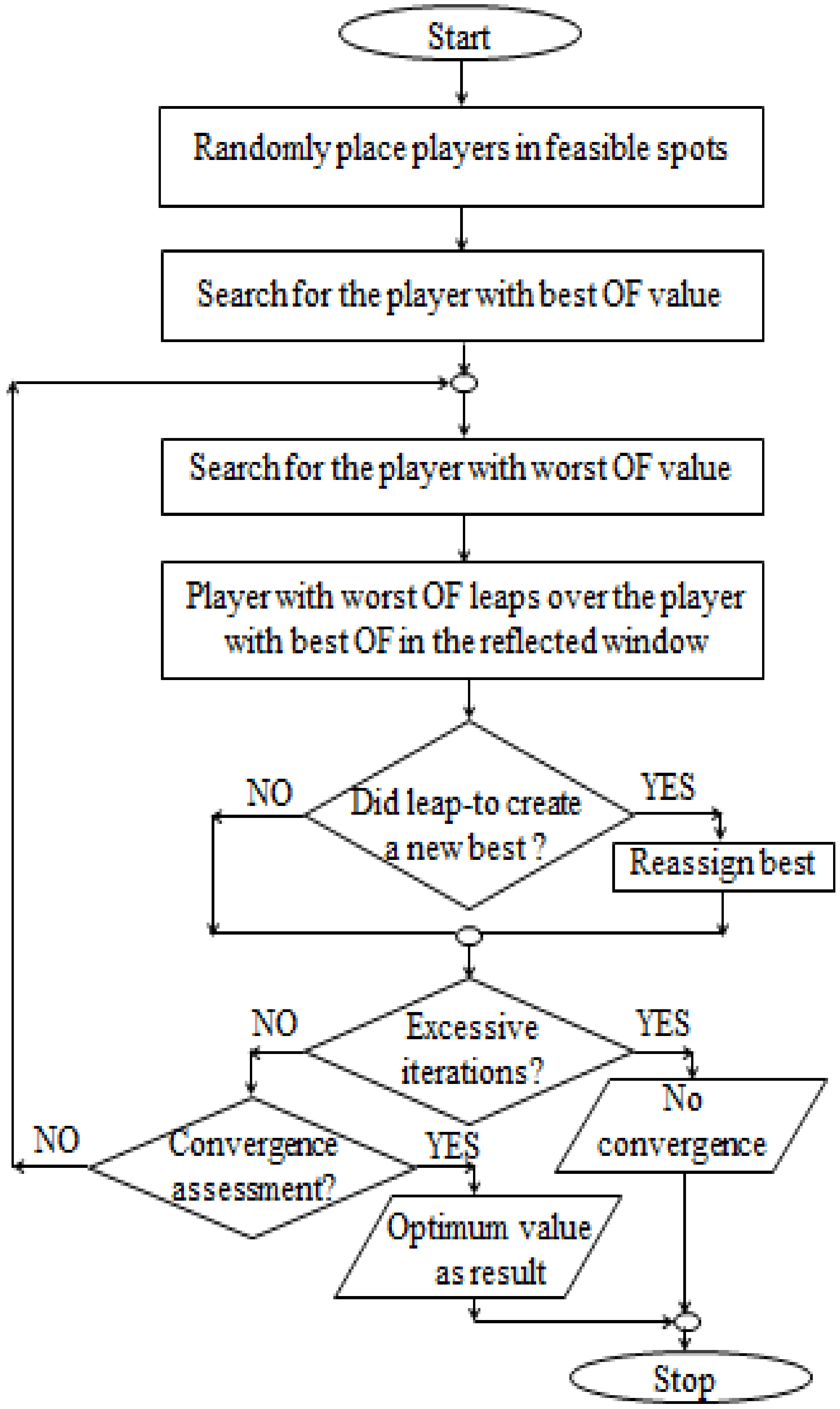

Start
Randomly place players in feasible spots
Search for the player with best OF value
Search for the player with worst OF value
Player with worst OF leaps over the player with best OF in the reflected window
Did leap-to create a new best ?
NO
YES
Reassign best
Excessive iterations?
NO
YES
No convergence
Convergence assessment?
NO
YES
Optimum value as result
Stop

Figure 2-2: Leapfrogging Flowchart

2.3 Optimizer Evaluation:

In general, the number of iterations required for the optimization technique to converge and reach a solution is used as a scale to compare different optimization techniques. But, unfortunately there is not a common quantification for an iteration. The way an iteration progresses changes considerably from one optimization technique to another. Hence a consistent technique was employed to evaluate the optimization techniques.

In order to evaluate an optimization technique, two major criteria act as competing primary objectives. The computational effort of the optimization technique can be directly accounted by computing the number of OF evaluations (NOFE). This quantifies the cost of evaluating and finding the solution of the optimization technique. The other competing criterion is the optimizer capability to find an accurate solution. Accuracy is determined by the probability of the solution being at the vicinity of the global optimum. Accuracy is computed as either the probability of finding/reaching the global optimum or as the probability of finding/reaching an OF which is at a desired proximity of the global optimum. Both these competing criteria are combined together as one metric. This is known as the probable number of function evaluations to converge at the global optimum (PNOFE).

The optimizers have to be run several times in order to avoid premature convergence or getting stuck at the local optimum. Iyer and Rhinehart [47] came up with a method to compute the number of independent trials required for an optimization technique to reach the global optimum or find an OF value that is at a close proximity to the global optimum.

$$N = \frac{\ln(1-c)}{\ln(1-f)} \tag{2-2}$$

Equation (2-2) is used to find the number of independent trials required. N represents the number of independent starts, f represents the best fraction which is the probability of finding an OF value that is at the vicinity of the global optimum with a confidence of c.

PNOFE is calculated from ANOFE and N given by Equation (2-3). ANOFE is the average of NOFE across all the trials.

$$\text{PNOFE} = \text{N} * \text{ANOFE} \tag{2-3}$$

In most of the cases, prior knowledge about the global optimum value is not present. Hence optimizers are initially run several number of times and the cumulative probability distribution function (CDF) against the OF values is plotted.

Figure 2-3 represents the CDF of an optimization technique. In this case 0.05 is the global optimum value (OF1*). From Figure 2-3 it can be inferred that the immediate next optimum value higher than 0.05 that is being found by the optimization technique is 0.1 (OF2*). The value of f Equation (2-2) represents the probability of finding the global optimum or the vicinity of the global optimum. While OF1* represents the global optimum, the vicinity of the global optimum is represented by Equation (2-4).

$$\text{OF} < \text{OF2}* \tag{2-4}$$

From Equation (2-4), it can be understood that any OF value that is lesser than the next optimum value but higher than the global is considered as being at the vicinity of the global optimum.

In Figure 2-3 the optimization technique could find the global optimum in about 30% of the total number of independent trials as 30 % of the times the optimization technique converged either at the global optimum or vicinity of the global optimum. Hence f in Equation (2-2) will be 0.3. If 0.1 is considered as being at the proximity of the global optimum then the optimization technique could find the optimum in about 97% of the total number of independent trials considered, hence f will be 0.97. If a person wants to be 99% confident that at least once in N trials the 0.05 OF value is found, the number of initial independent trials (N) is computed using Equation (2-5):

$$N = \frac{\ln(1-0.99)}{\ln(1-0.3)} \approx 13 \tag{2-5}$$

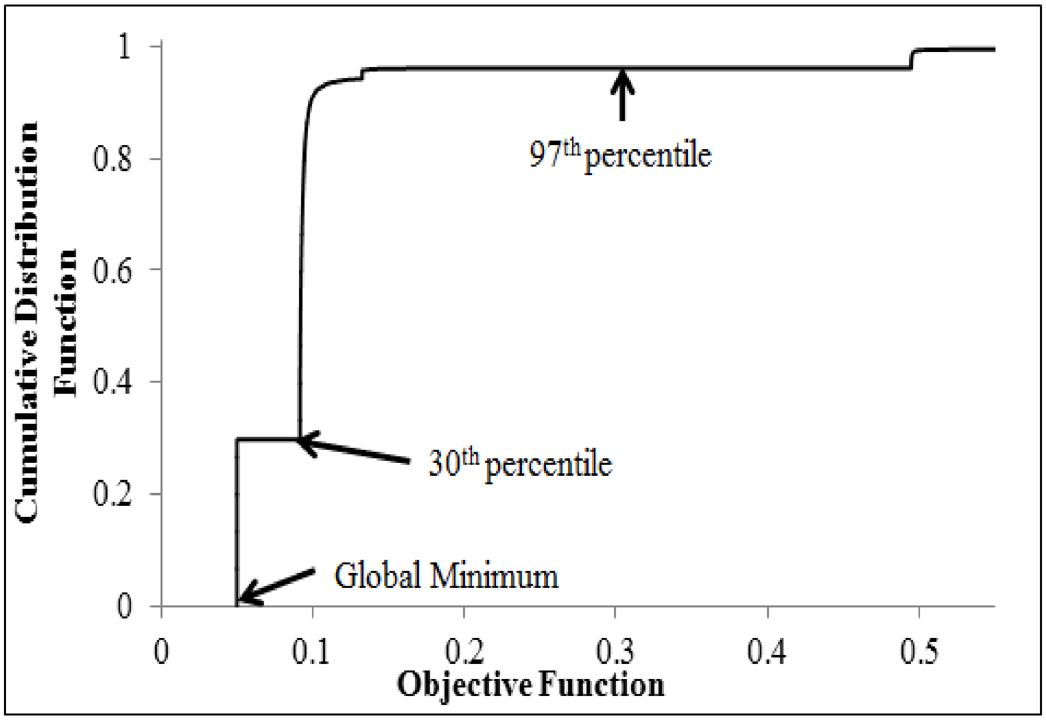


Figure 2-3: Cumulative Distribution Function (CDF)

So with 99% confidence it can said that one among 13 trials will find the global optimum value. If ANOFE is 437.6 then PNOFE is computed as shown in Equation (2-6).

$$\text{PNOFE} = 13 * 437.6 \approx 5689 \tag{2-6}$$

Apart from these, simplicity of the optimization technique, robustness of it to handle surface aberrations and discontinuities and scalability of the optimization technique to higher dimensions are considered as secondary criteria for evaluating the optimization techniques.

2.4 Evaluation of Leapfrogging and Other Optimization Techniques:

The PNOFE of the various optimization techniques over different functions are compared. Table 2-1 is sourced from the publication on Leapfrogging optimization technique [1].

Table 2-1: Effort per Benefit (PNOFE) of Optimization Techniques

	OF value	**Optimizer (PNOFE)**			
Function ↓		HJ	LF	PS	RLM
Bootprint with pinhole	Global	95,800	9,820	22,100	577,000
Sharp valleys with flat well	0.05	Infinity	2,960	Infinity	Infinity
Bootprint with constraint	0.2257	7,150	2,550	6,350	321,000

Table 2-1 lists the PNOFE values of Hook Jeeves (HJ), Leapfrogging (LF), Particle Swarm (PS) and RLM (Modified Levenberg Marquardt) methods. These optimizers are tested over three different functions that exhibit surface aberrations. Bootprint with pinhole has a global optimum that is not easily discovered, and it can be seen from the table that Leapfrogging has the least PNOFE among all the optimizers. Sharp valleys with a flat well function has a flat surface that leads to a well. RLM, which is a gradient based technique resulted in 0 values of the gradient and Hessian and so the optimizer was stuck and could not find the global. Neither PS nor HJ, though are direct search techniques, were successful because of their difficulty in crossing the flat well. Leapfrogging once again did well on this function. Bootprint with constraint function has a circular constraint that divides and creates two local optima close to the global optimum. Once again it can be seen that the Leapfrogging has the lowest PNOFE value. From the table it can be inferred that Leapfrogging performed better than the other optimizers.

While Table 2-1 compared the optimizers based on PNOFE, a ccomparison based on a secondary criterion was also performed. Table 2-2 is a subjective comparison of the optimization techniques sourced from the publication on Leapfrogging optimization technique [1].

Table 2-2: Ranking of Complexity Factors

Criteria/Optimizer	**HJ**	**LF**	**PS**	**RLM**
Writing and debugging the code	2	1	3	4
Scaling to high dimensions	1	1	1	2
Number of loops used	1	2	2	1
Work during each iteration	2	1	3	2
Amount of data stored	1	2	2	1
Sum of all attribute ranks	**7**	**7**	**11**	**10**
Overall rank	1	1	3	2

It can be seen from Table 2-2 that Leapfrogging and HJ both secured rank 1 based on the complexity factor analysis. Combining the analysis of Table 2-1 and Table 2-2, it can be seen that Leapfrogging performed better than other optimizers.

2.5 Advantages of the Leapfrogging Technique:

The advantages of Leapfrogging optimization technique are:

1. It is a direct search technique
2. It is robust to surface aberrations
3. It exhibits lowest effort to benefit (PNOFE) value
4. It can handle constraints
5. It is user friendly and
6. It is scalable to higher dimensions

2.6 Leapfrogging Applications:

So far Leapfrogging has been employed for a wide range of applications. Here is list of the applications:

1. Process model based controller on HX process for comparison against conventional controllers [2].
2. Several studies for modeling viscoelastic properties [1, 3-6].
3. Study of incorporation of TSK models in model predictive control [7].
4. Several Unit Operations Lab studies, rheology modeling and academic performance prediction modeling at OSU.

2.7 Horizon Predictive Control [48-55]:

Traditional proportional-integral-derivative controllers or advanced process controllers are used for most of the industrial control applications. This is because of their credibility and ease of implementation. Since most of the processes in chemical industries are nonlinear and multivariable, advanced process controllers (multivariable controllers) would be preferred more than traditional controllers.

HPC is being successfully used and preferred in the process industry for various applications since 1980. Starting from the end of 1970s various articles on HPC expressed the interest of their usage in the industries [22, 48]. This controller algorithm controls the action of the controlled variable (CV) at the current time instant while keeping the future action in mind. This controller continuously estimates the CV action in future, which is initialized at the present time instant. The objective of this controller is to predict the control actions (Manipulated Variable (MV) values) that should be employed in the future by minimizing the future predicted actuating error (CV from set point (SP)) within operational restrictions [56]. MV is the input variable that is dynamically adjusted to keep the CV at the SP. This project will implement a nonlinear horizon predictive control on the HX process.

Figure 2-4 shows a schematic of HPC adapted from [51]. In order to implement this controller, the process dynamic response is first modeled. In general, the true process behavior is unknown and model is used as a proxy for the process. To develop a model, first, process data are collected by conducting experiments. Then, optimization techniques are used to find the best set of model parameters that make the model match the process.

An optimization algorithm is also included in the controller algorithm to forecast an MV sequence to reduce the sum of squared deviation between the CV and the process SP. The optimization gives the set of control actions, which is represented by the sequence of future MV moves to the process model. The process model returns the model output for the set of control action provided by the optimization process. This flow of data between the optimization process and the model will continue until the optimization technique finds the best set of control actions.

This optimization technique while keeping the future CV action in mind, should minimize the CV deviation from the SP as shown in Equation

$$\min \sum (SP - CV)^2 \tag{2-7}$$

Hence the implementation of HPC is computationally demanding [57]. Multiple future moves of the MV are computed by the optimization algorithm at every time instant to make the CV follow the reference trajectory, along a desired path, and avoiding constraints. Reference trajectory is a recommended path by which the CV reaches the SP. The first MV value of the multiple MV moves the optimization technique suggests is given as CV to the process. This project will calculate 3 future moves and of the three MV moves only the first move is implemented.

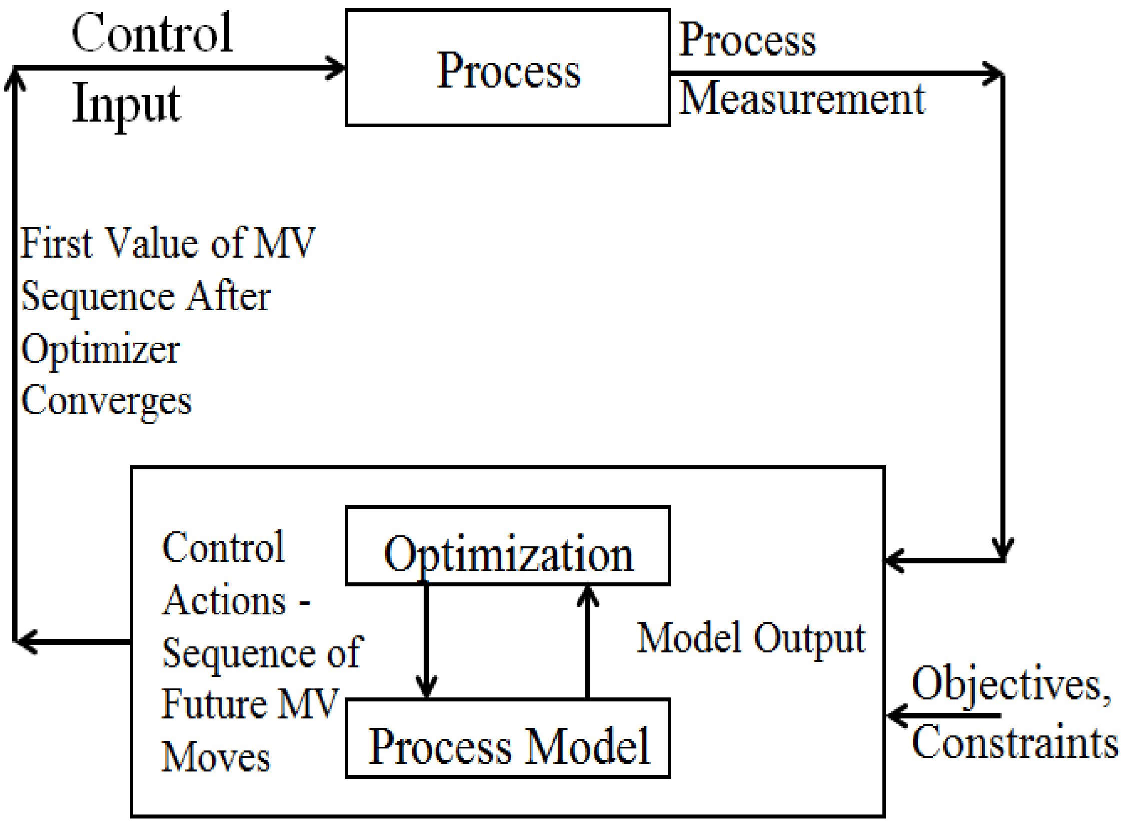


Figure 2-4: Horizon Predictive Control Schematic – Adapted From [51]

Since these controllers are used for industrial applications, optimization technique used within the controller algorithm should be computationally fast, should be able to handle surface aberrations and discontinuities, and should be able to find the global optimum when there is a possibility for the presence of local optimum. A direct search based optimization technique is better in these cases. Since Leapfrogging possesses all the required qualities, it can be used in HPC. The successful implementation of Leapfrogging in HPC was already demonstrated [7].

2.8 Process Modeling [56]:

A dynamic model of the process has to be created in order for the optimization in the HPC to use for finding the best set of control actions. In this study a nonlinear model of the HX process is developed.

The process model employed in the HPC should capture the the process dynamics so that it can predict the behavior of the process for the optimizer to fulfil its responsibility. At the same time the model has to simple and user-friendly.

There exist several methods to develop nonlinear process models. In this study, first principles modeling of the HX process will be employed. To collect process values across a wide range of operating conditions, and to find the best set of model parameters that make the model match the process, a skyline function was established by changing the MV and collecting the process values.

To find the best set of model parameters that make the model capture the trend in the process an optimization technique has to be employed. In this study Leapfrogging optimization technique will be employed to find the best set of model parameters.

The improved Leapfrogging optimization technique developed through fundamental analysis will also be used to find the best set of model parameters that minimizes the process model mismatch. The effort to benefit (PNOFE) of both the optimization techniques will be compared in this study.

2.9 OF for HPC:

The nonlinear first principles process model has to follow a path in the HPC algorithm to make the process move towards the set point. This path is called the reference trajectory. The OF to the optimizer within the HPC is to minimize the distance between the model and the reference trajectory within the horizon. Horizon is the time in future over which the behavior of the model (proxy for the process) is considered to find the best control action to be taken at the current instant. The reference trajectory path is computed using the required set point and the process model mismatch. If the process model follows the reference trajectory, the actual process will reach the set point.

In this project Leapfrogging and its improvement will be implemented as a part of nonlinear HPC to demonstrate their applicability and establish their credibility.

CHAPTER 3

METHODOLOGY

The methodology for each of the specific aims discussed in Chapter 1 (Introduction) is discussed below.

3.1 Improved Initialization Methodology:

The following section discusses the methodology for Aim 1: Provide fundamental knowledge and further improve efficiency of the technique [8].

Leapfrogging optimization technique as discussed in Chapter 2 (Background) proceeds as follows:

1. Players are randomly initialized on the feasible DV space
2. Best and worst players are found
3. Worst player leaps over the best player into the reflected hyper-volume
4. Best and worst players are reassigned
5. Convergence assessment (stopping criteria) is tested
6. Repeat 3, 4 and 5 until the convergence assessment is true

Let the OF value of the best player be OF_B and the OF value of the worst player be OF_W. During every leap over the worst player leaps into the reflected hyper-volume connecting the best and the worst player. All the other players remain in their original locations. Only the worst player moves during every leap over, so the OF value at the new spot is evaluated after the leap over. Let the OF value of the new spot be OF_{NW}. Since all the other players remain in their same locations, the OF_B is compared with OF_{NW} to find the new best player. Equations (3-1), (3-2) and (3-3) represent the possible results of the comparison between OF_B and OF_{NW}. Generally after a leap over if the player finds a spot at the constrained region, it leaps over into the reflected hyper volume connecting the player itself and the previous best player. This process will continue until the player leaps into a spot which is not in the constrained region. Hence, in all the 3 possible results below OF_B is assumed to be calculated at a spot where the constraint is "OK".

$$OF_B < OF_{NW} \tag{3-1}$$

$$OF_B = OF_{NW} \tag{3-2}$$

$$OF_B > OF_{NW} \tag{3-3}$$

In the case of Equation (3-1), since OF_B is smaller than OF_{NW}, the player with OF value OF_B (previous best player) will continue to be the best player for the next leap over. The concept is shown in Figure 3-1. Figure 3-1 is an example of 1 dimension function. The worst player leaps over to a spot where the OF value OF_{NW} is greater than the best player OF_B. Hence OF_B remains the best player.

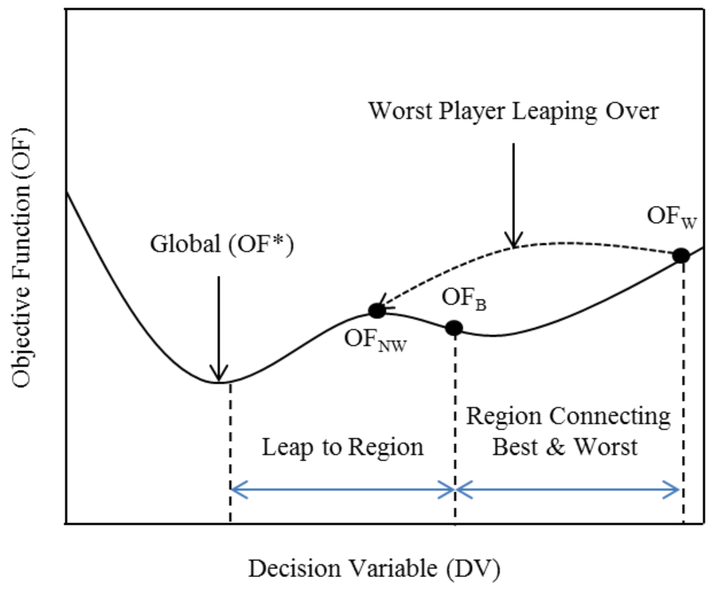


Figure 3-1: Best Player Remains Best After Leap Over

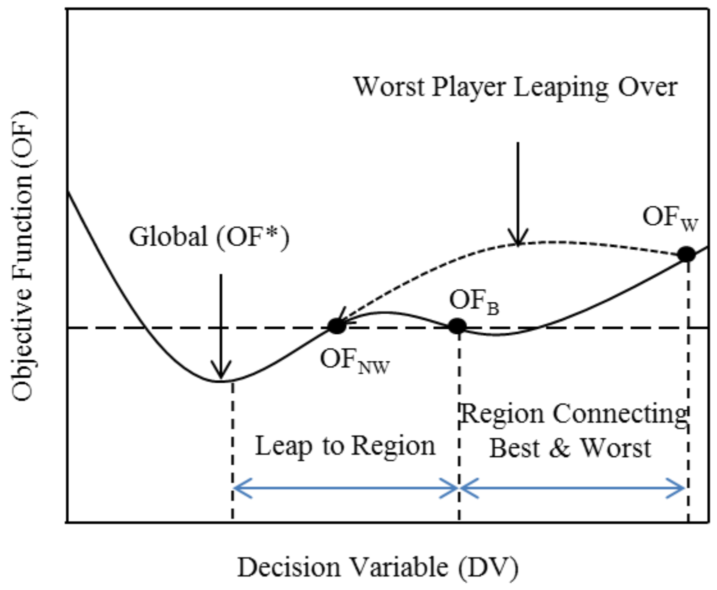


Figure 3-2: OF At Leap-to Spot Same As Best

In the case of Equation (3-2), since OF_B is equal to OF_{NW}, either the player with OF value OF_B (previous best player) or the player with OF value OF_{NW} can be regarded as the best player for the next leap over. In order to keep the algorithm of the Leapfrogging optimization technique simple, the previous best player will continue as the best player for the next leap over. Per the Leapfrogging algorithm if two players have the same OF value which is lower than all the other players, then the first player to find the OF value among the two players is considered as the best player. Figure 3-2 shows the concept in a 1 dimension function example. The dotted line connecting the points at which the OF values are OF_{NW} and OF_B shows that the OF at these two points are the same.

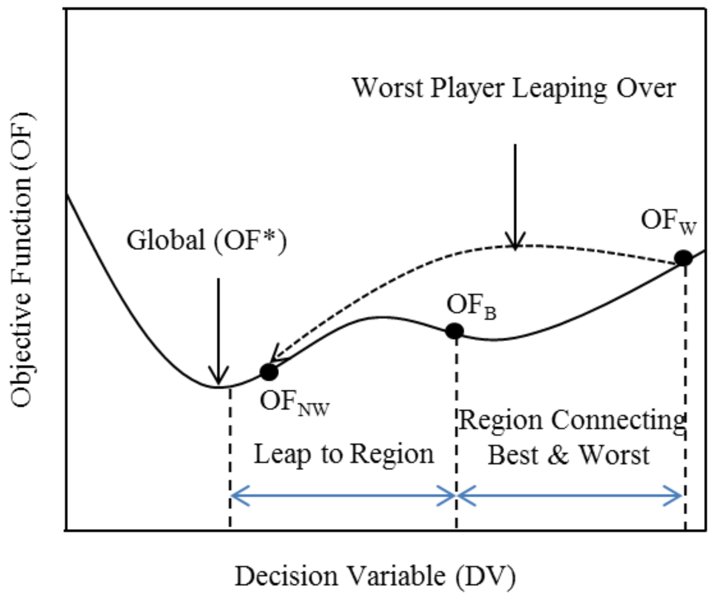


Figure 3-3: Leap-to Spot Better Than Best

In the case of Equation (3-3), since OF_{NW} is smaller than OF_B, the player with OF value OF_{NW} (new spot of the leapt over worst player) will be considered as the best player for the next leap over. Figure 3-3 shows this case on a 1 dimension example. The OF value at the leap to spot

OF_{NW} is better than OF_B and hence the new player (leapt over worst player) is regarded as the best player for the next leap over.

To sum up, the best player with OF value OF_B will remain the best player until and unless a worst player leaps over and finds a spot where the OF value OF_{NW} is better than OF_B. If a player during random initialization is present at the vicinity of the global (OF value of the player < OF2*, where OF2* is the immediate next optimum value greater than the global optimum OF* as explained in Equation (2-4)), then that player will be regarded as the best player until a worst player leaps over and finds a better spot. Since the best player with OF value OF_B is already in the vicinity of the global optimum, if the leapt over worst player with OF value OF_{NW} finds a better spot as shown in Equation (3-3), the leapt over player will also be in the vicinity of the global optimum i.e. $OF_{NW} < OF2^*$.

Hence, if a player during random initialization is present at the vicinity of the global, then during successive leap overs it will attract all the other players towards the vicinity of the global leading to a final convergence at the global optimum or the vicinity of the global optimum. This fundamental knowledge is used for improving the initialization of the player by which at least one player is present at the vicinity of the global at the end of the initialization stage.

The idea behind this improvement is to

1. Explore the surface with greater number of players
2. Sort the players based on the OF values
3. Pick the top (M*10) players for proceeding with Leapfrogging optimization

The plan is to explore the feasible DV space first with greater number of players. This would increase the probability of placing at least one player in the vicinity of the global optimum, which would eventually attract all the other players towards it. Then sort the OF values of all the players and pick the top set of players to proceed with the Leapfrogging optimization technique.

Based on experience it has been found that there should be at least 10 players per dimension to avoid premature convergence or on the side of a hill. If there are a total of M dimensions then at least (M*10) players are required for carrying out the Leapfrogging optimization as successfully as expected.

An increase in the number of players in the initial exploration will lead to greater NOFE (number of function evaluations – one function evaluation corresponds to one computation of the OF value), and hence there needs to be balance between increasing the benefit and the corresponding effort involved.

Figure 3-4 shows the Jupitor's Eye function scaled from 1 to 10 on both the DVs and the center of the inner most contour is the global optimum marked by a cross. This is an example of a single optimum function.To carry Leapfrogging optimization technique on the Jupitor's Eye function 20 players are placed randomly on the entire surface of the contour, as shown in Figure 3-4.

Figure 3-5 shows the same Jupitor's Eye function contours, but with improved initialization. Initially the surface is explored by 4600 random players. The 4600 players are sorted based on their OF values and top 20 players are then used for the Leapfrogging optimization technique. The choice of 4600 players for initial surface exploration is done by using 0.99 for c and 0.001 for f_A in Equation (3-16) discussed in the next section of this chapter. By initial surface exploration it can be seen from Figure 3-5 that even before beginning the leap overs, the players have surrounded the global optimum

Figure 3-6 shows the original Leapfrogging initialization with Peaks function scaled from 1 to 10 on both the DVs. Peaks is a multiple optimum function which has three well-shaped minima. The global optimum is marked by a cross. It can be seen from Figure 3-6 that all the 20 players are placed randomly all over the entire DV space.

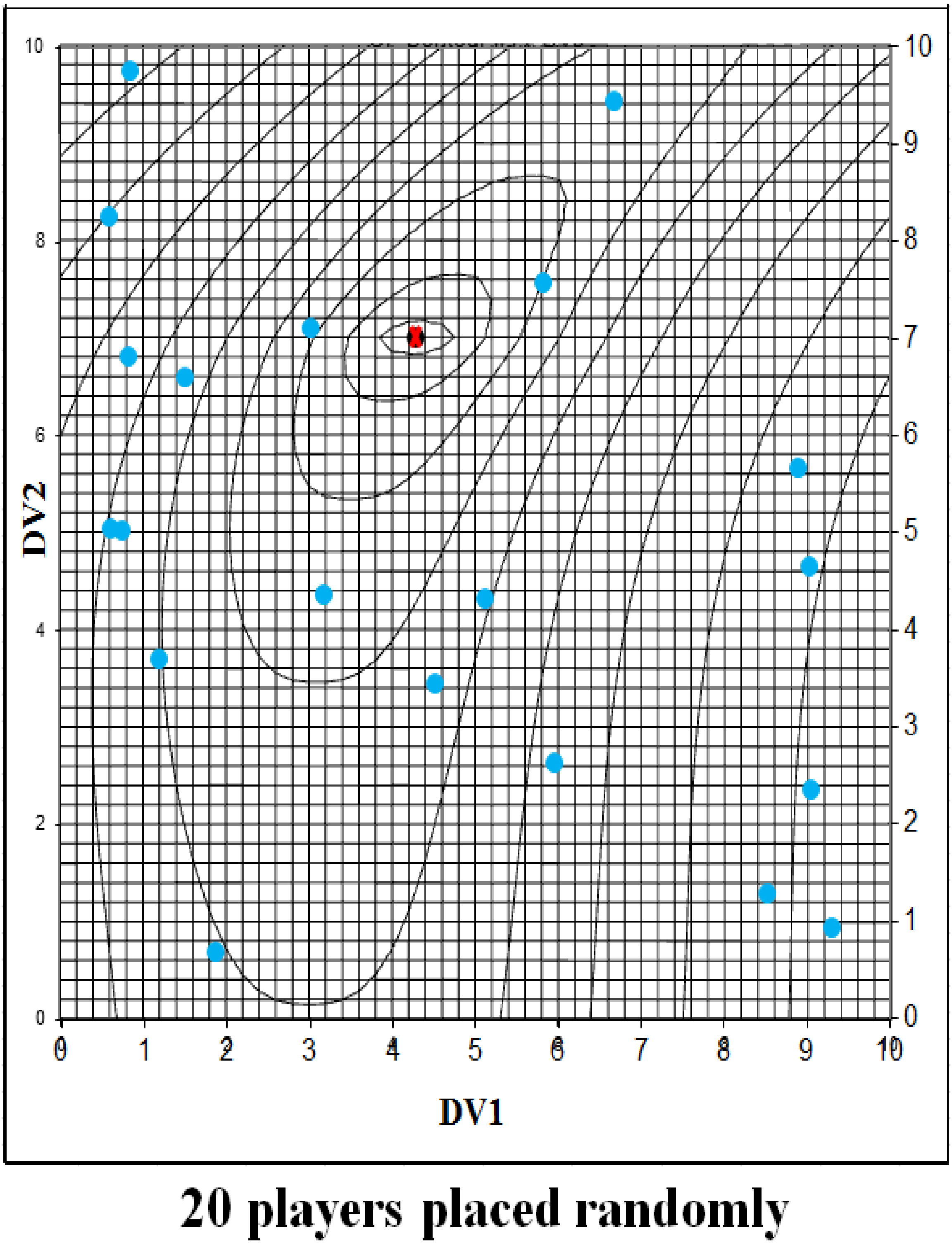
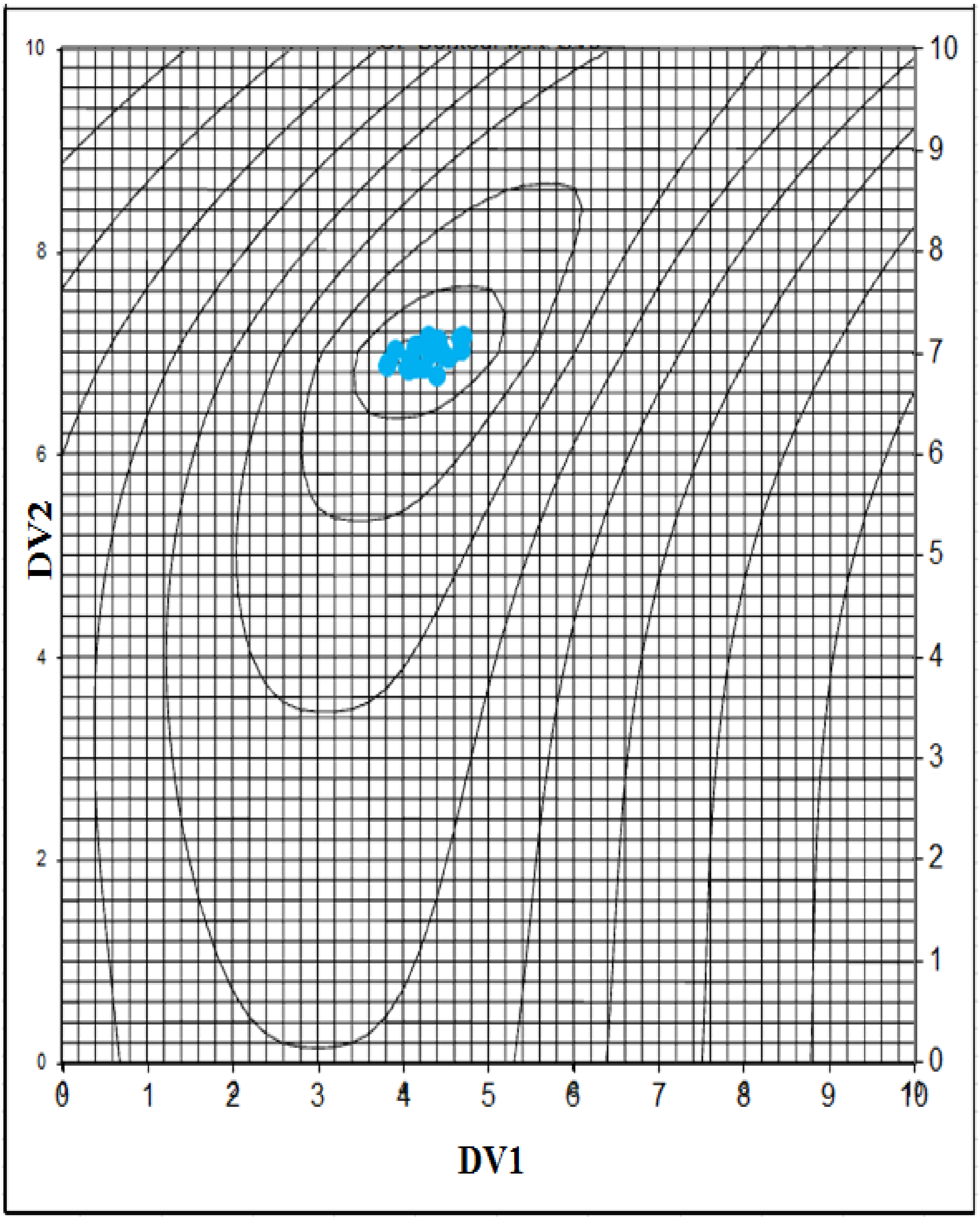


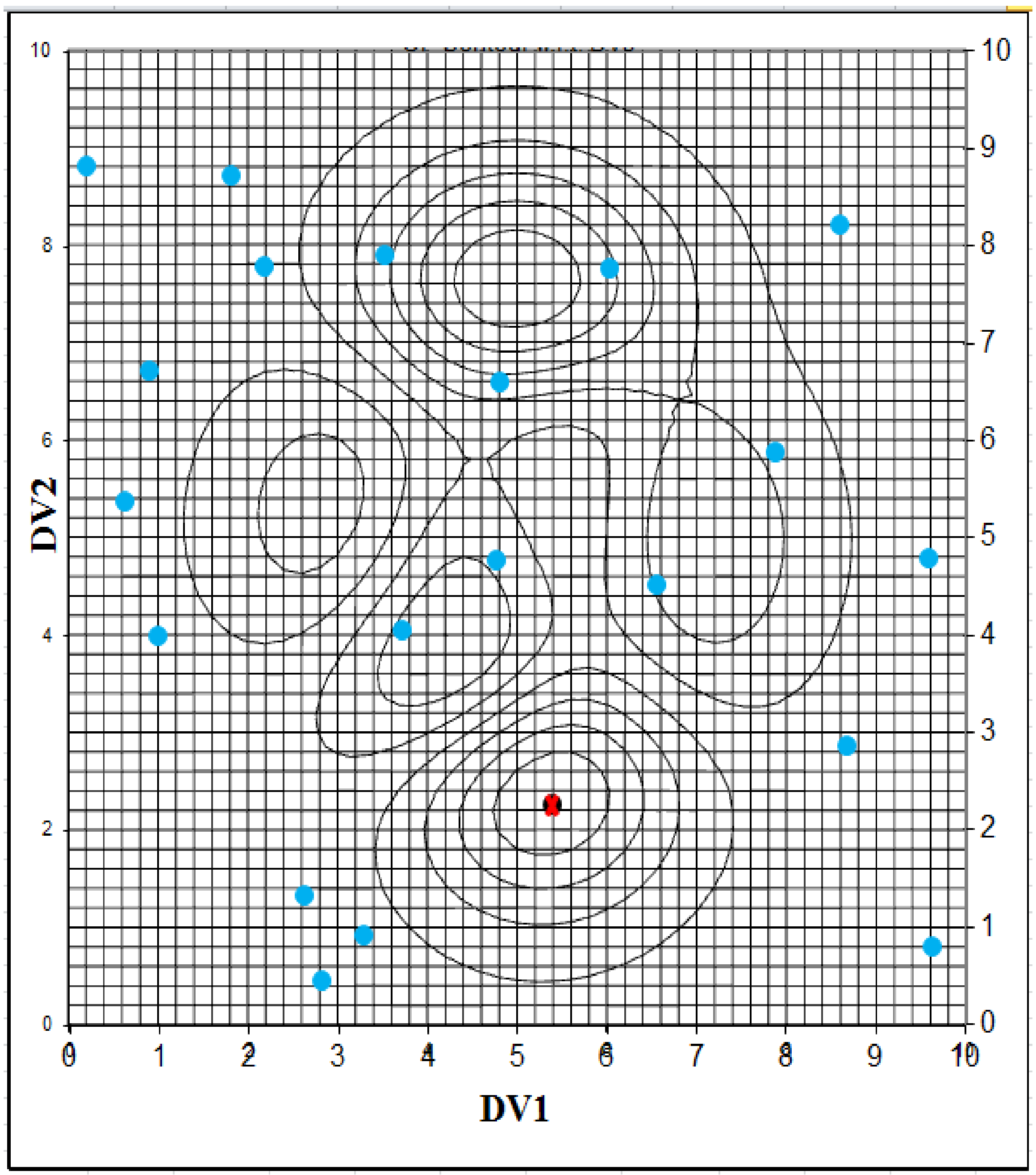
Figure 3-4: Initialization with Original Leapfrogging Optimization Technique



Top 20 of 4600 players placed randomly

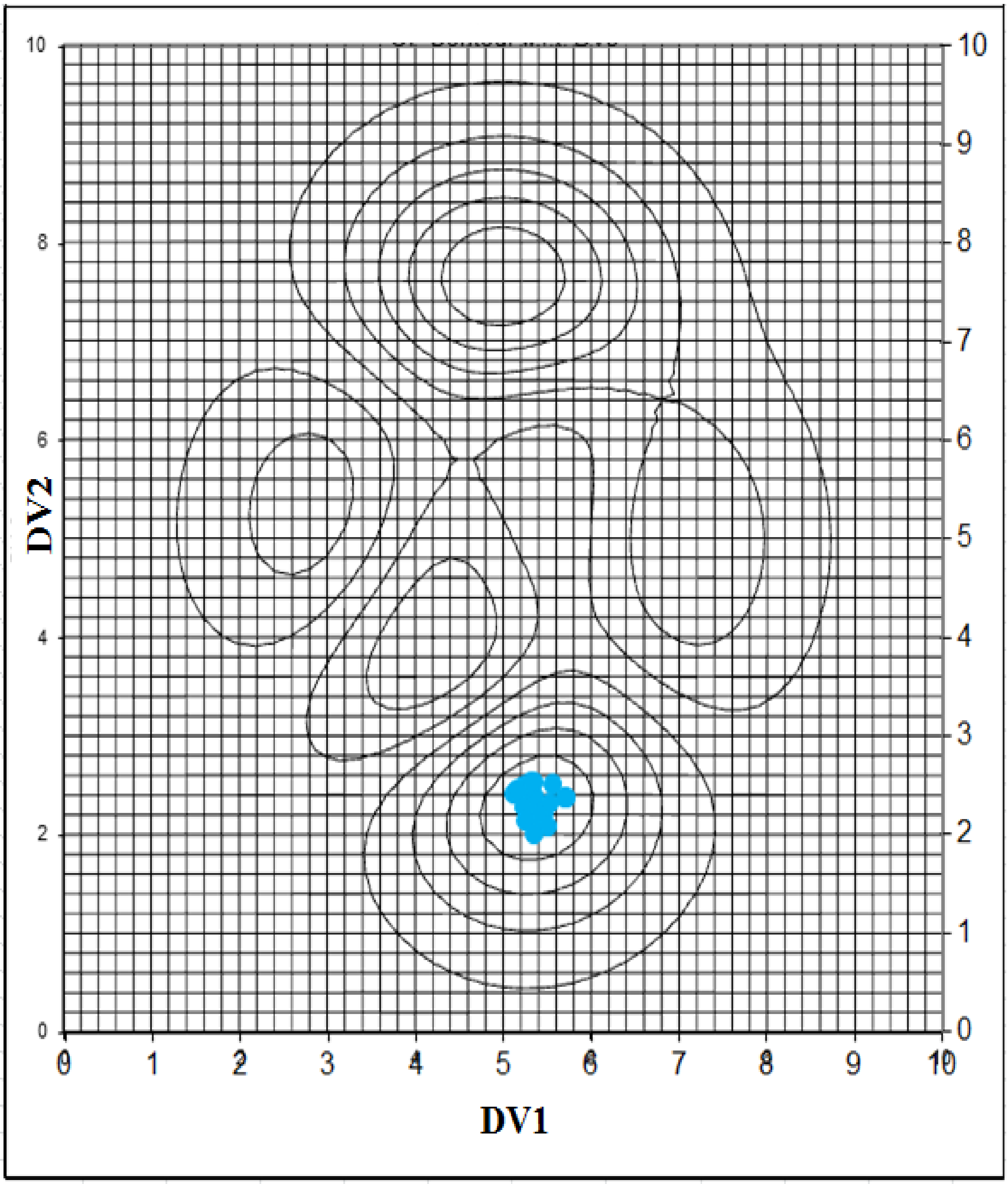
Figure 3-5: Initialization with Improved Leapfrogging Optimization Technique

.



20 players placed randomly

Figure 3-6: Initialization with Original Leapfrogging Optimization Technique



Top 20 of 4600 players placed randomly

Figure 3-7: Initialization with Improved Leapfrogging Optimization Technique

Figure 3-7 shows the same Peaks function contours, but with improved initialization. Initially the surface is explored by 4600 random players. The 4600 players are sorted based on their OF values and top 20 players are then used for the Leapfrogging optimization technique. If by initial surface exploration 4600 players are placed and the top 20 players are picked, it can be seen from Figure 3-7 that even before beginning the leap overs, the players have surrounded the global optimum even in the case of a multiple optima function.

Figure 3-5 and Figure 3-7 show that in the case of both, single optimum and multiple optima functions, improved initialization helps surrounding the players around global optimum during the initialization stage of the Leapfrogging optimization technique. The expectation from the above is that the improved initialization would lead to the global optimum much faster than the original version.

3.1.1 Mathematical Methodology:

The improved initialization is built on the fact that at least one player present in the vicinity of the global optimum will attract all the other players towards it. Through improved initialization, the best player will anyway be in the vicinity of the global optimum and if a better spot is found, it would be much closer to the global optimum. Vicinity of the global optimum is defined as an OF value that is lesser than the OF values at any of the other local optimum (Equation (2-4)).

Figure 3-8 demonstrates the concept of the global attractor area. Figure 3-8 shows in 1 dimension the presence of local and global minimum. On the X-axis is the DV, and on the Y-axis is the corresponding OF value. The global attractor area is presented by Region A (hatched region). In Region A, the OF value will be lower than the local optimum values. So, if at least one player is present in the Region A, then it would attract all the other players towards the global optimum.

Mathematical analysis has to be established on the initial number of players that are required for surface exploration so that the confidence in having at least one player in the proximity of the global optimum is desirably high.

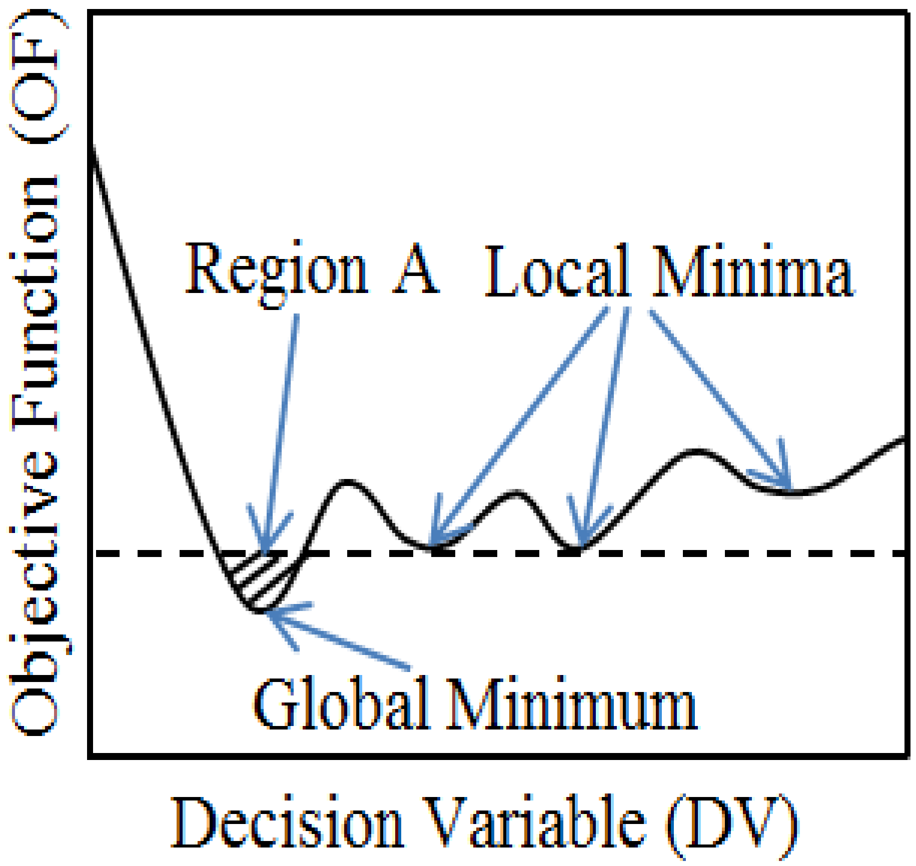


Figure 3-8: Concept of Global Attractor Area

The desire is to find the number of initial players that are required for the surface exploration to provide a desired confidence that at least one player during the initialization stage will have an OF value less than any other local optimum (OF < OF2*). This can be established through simple probability analysis shown in Equation (3-4):

$$
\begin{aligned}
P(at\ least\ 1\ of\ M_i\ Players\ in\ A) &= 1 - P(none\ of\ M_i\ Players\ in\ A) \\
&= 1 - P(1^{st}\ not\ in\ A,\ \underline{and}\ 2^{nd}\ not\ in\ A,\ \underline{and}\ .....) \\
&= 1 - P(any\ one\ player\ not\ in\ A\ )^{M_i} \\
&= 1 - [1 - P(any\ one\ player\ in\ A\ )]^{M_i} \qquad (3\text{-}4)
\end{aligned}
$$

The set of at least 1 of M_i initial players used for surface exploration being in Region A and the set of none of M_i players being in Region A are mutually exclusive and exhaustive. Hence the probability of both the sets add up to 1. Therefore, the probability of at least 1 of M_i players in Region A can be computed as the difference between 1 and the probability of none of the M_i players in Region A.

Expanding on the fact that the probability of none of the M_i players are in Region A leads to the probability of the 1st player not in Region A and 2nd player not in Region A and so on and so forth. In probability theory "AND" represents the probability of the union of the two sets. Probability of union of two sets is same the product of probabilities of the individual sets. This idea is further used to simplify the probability obtained in the Stage 2 of Equation (3-4).

So, probability in Stage 2 of Equation (3-4) is further simplified as the product of probabilities of 1st player not in Region A, 2nd player not in Region A and so on and so forth. The probability of every single player not in Region A is equi-probable. Hence, the product of the probability of every player not in Region A can be computed as the probability of any one player not in Region A raised to the power of number of initial players for surface exploration, $M_{i.}$

The set of any one player not in Region A and any one player in Region A are exclusive and exhaustive, hence the sum of probabilities of the two sets viz. probability of any one player in Region A and any one player not in Region A add up to 1. So, the probability of any one player not in Region A is the difference between 1 and probability of any one player in Region A. To sum it up probability of at least one of the M_i players to be in Region A is a function of the probability of any one player in Region A as shown in Equation (3-4).

Equation (3-4) is rearranged to obtain Equation (3-5) which provides a mathematical equation to compute for the number of initial players required for surface exploration.

$$M_i = \frac{\ln[1 - P(at\ least\ 1\ of\ \ M_i\ \ Players\ in\ A)]}{\ln[1 - P(any\ one\ \ player\ in\ A\ )]} \tag{3-5}$$

The probability of any one player to be in the global attractor region A is represented by $f_{A.}$ The probability of at least one of M_i initial players to be in Region A is the confidence with which at least one of the M_i initial players is in Region A at the end of the initialization stage can be said. The confidence is represented by c. Hence Equation (3-5) is represented as Equation (3-6).

$$M_i = \frac{\ln(1 - c)}{\ln(1 - f_A)} \tag{3-6}$$

Equation (3-6) is of a similar format as (2-2). In Equation (3-6), c represents the confidence that atleast 1 player is in the Region A during initialization while in Equation (2-2), c represents the confidence with which the optimizer converges at the vicinity of the global optimum. In Equation (3-6) f_A stands for the fraction of Region A in the total DV space while in Equation (2-2) f represents the probability of converging at the vicinity of the global optimum.

While the initialization technique provides a high confidence that at least one player is in the vicinity of the global optimum, Leapfrogging optimization technique as such can fortuitously leap a player into the Region A even if the initialization did not result in one player in the Region A. This implies that the probability of reaching the global optimum will be greater than the expected value as one player being in the vicinity will progress towards the vicinity of the global optimum in successive leap overs and not get converged at any of the local optimum.

Two expectations will be tested in this study. The improved initialization provides:

1. Greater probability of finding the global optimum OF* than expected because of the possibility of players being placed in A fortuitously during leap overs, and
2. Reduction in PNOFE because of the greater probability of finding the global optimum OF*.

3.2 HX Model Development:

The following section discusses the methodology for Aim 2: Apply original and improved Leapfrogging on nonlinear pilot scale HX modeling. The aim is to test the applicability of original Leapfrogging optimization technique and the Leapfrogging optimization technique with improved initialization in modeling a pilot scale HX and compare the results.

The HX 001 of the Unit Operations Laboratory at OSU, Stillwater was considered for this study.

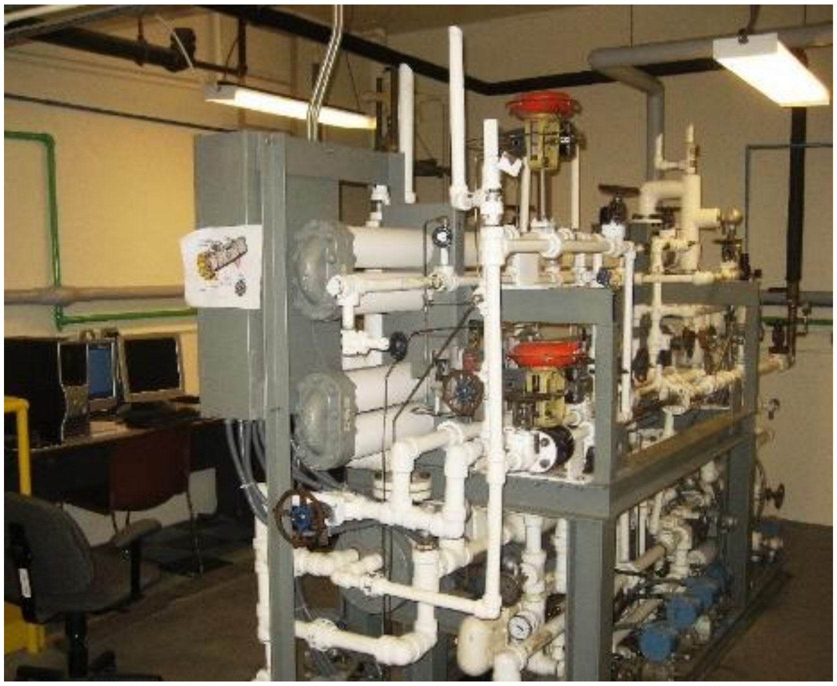
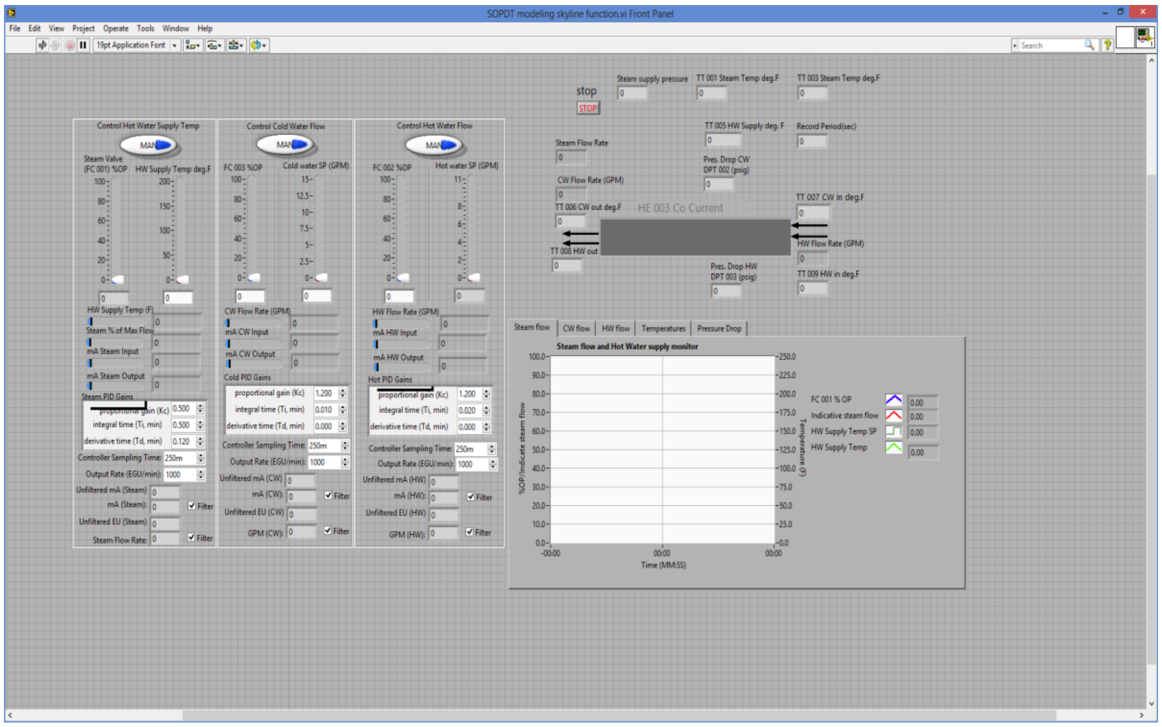


Figure 3-9: HX Network at Unit Operations Lab, OSU

Figure 3-9 is a picture of the HX network present at the Unit Operations Lab at OSU, Stillwater. The HX network consists of 6 HX units; 3 Shell and Tube HXs, 2 Double Pipe HXs and 1 Plate and Frame HX. For this study a Shell and Tube HX (HX001) is considered. The HX 001 is a four-pass Shell and Tube heat exchanger with steam on the shell side and water on the tube side. Steam is used to increase the temperature of the water on the cold side. Figure 3-10 shows the graphic user interface (GUI) of the HX001. LabView simulator is used for building the graphic user interface.



Steam supply pressure 0
TT 001 Steam Temp deg.F 0
TT 003 Steam Temp deg.F 0
TT 005 HW Supply deg. F 0
Record Period(sec) 0
Steam Flow Rate 0
Pres. Drop CW DPT 002 (psig) 0
CW Flow Rate (GPM) 0
TT 006 CW out deg.F 0
TT 007 CW in deg.F 0
HW Flow Rate (GPM) 0
TT 008 HW out 0
Pres. Drop HW DPT 003 (psig) 0
TT 009 HW in deg.F 0

Steam Valve (FC 001) %OP 0
HW Supply Temp deg.F 0
HW Supply Temp (F) 0
Steam % of Max Flow 0
mA Steam Input 0
mA Steam Output 0
Steam PID Gains
proportional gain (Kc) 0.500
integral time (Ti, min) 0.500
derivative time (Td, min) 0.120
Controller Sampling Time: 250m
Output Rate (EGU/min): 1000
Unfiltered mA (Steam) 0
mA (Steam): 0 ✓ Filter
Unfiltered EU (Steam) 0
Steam Flow Rate: 0 ✓ Filter

FC 003 %OP 0
Cold water SP (GPM) 0
CW Flow Rate (GPM) 0
mA CW Input 0
mA CW Output 0
Cold PID Gains
proportional gain (Kc) 1.200
integral time (Ti, min) 0.010
derivative time (Td, min) 0.000
Controller Sampling Time: 250m
Output Rate (EGU/min): 1000
Unfiltered mA (CW) 0
mA (CW): 0 ✓ Filter
Unfiltered EU (CW) 0
GPM (CW): 0 ✓ Filter

FC 002 %OP 0
Hot water SP (GPM) 0
HW Flow Rate (GPM) 0
mA HW Input 0
mA HW Output 0
Hot PID Gains
proportional gain (Kc) 1.200
integral time (Ti, min) 0.020
derivative time (Td, min) 0.000
Controller Sampling Time: 250m
Output Rate (EGU/min): 1000
Unfiltered mA (HW) 0
mA (HW): 0 ✓ Filter
Unfiltered EU (HW) 0
GPM (HW): 0 ✓ Filter

FC 001 % OP 0.00
Indicative steam flow 0.00
HW Supply Temp SP 0.00
HW Supply Temp 0.00

Figure 3-10: HX001 GUI

The model is developed for use in implementing HPC on HX001. For the purpose of implementation of HPC, the outlet temperature of the cold side fluid is considered as the controlled variable (CV) and the steam valve opening on the hot side is considered as manipulated variable (MV). Hence the CV-MV relation is modeled.

First, the steady-state energy balance of the HX001 is established by the ideal concepts in Equation (3-7).

$$\text{Heat lost by steam} = \text{Heat gained by water} \tag{3-7}$$

Equation (3-7) represents a simple energy balance assuming there are no ambient losses and all the heat lost by the steam is gained by the water. This energy balance is represented in Equation (3-8).

$$\dot{Q}_{\text{lost by steam}} = \dot{Q}_{\text{gained by water}} \tag{3-8}$$

In Equation (3-8), $\dot{Q}$ is the heat of the representation in the suffix. The heat lost by the steam can be expressed as shown in Equation (3-9).

$$\dot{Q}_{\text{lost by steam}} = F_s\lambda \tag{3-9}$$

In Equation (3-9), F_s represents the mass flow rate of the steam and λ represents the heat of vaporization of steam. For the modeling purpose it is assumed that there is no sub cooling and λ is a constant independent of temperature and pressure. The valve used for controlling the flow rate of steam into the heat exchanger is an equal percentage valve. Hence the power law model [58] is used to relate the mass flow rate and the valve opening represented by Equation (3-10).

$$F_s = aX_s{}^b \tag{3-10}$$

In Equation (3-10), X_s represents the steam flow valve opening, a and b are the parameters of the power law model. The pressure drop is assumed to be a constant. Similarly the heat gained by water is given by:

$$\dot{Q}_{\text{gained by water}} = F_w\rho C_p(T_o - T_i) \tag{3-11}$$

In Equation (3-11), F_w represents the mass flow rate of water, ρ represents the density of water, C_p represents specific heat of water, T_o represents the outlet temperature of water and T_i represents the inlet temperature of the water. The density and the specific heat of water are assumed to be constant.

Substituting Equation (3-10) in Equation (3-9) and equating it to Equation (3-11) gives:

$$aX_s^{\,b}\lambda = F_w \rho C_p (T_o - T_i) \tag{3-12}$$

Equation (3-12) is rearranged to obtain a relation for the outlet temperature T_o (CV) as a function of the steam valve opening X_s (MV) represented by Equation (3-13)

$$T_{oSS} = T_i + \frac{\lambda a}{F_w \rho C_p} X_s^{\,b} \tag{3-13}$$

Equation (3-13) is the ideal steady state nonlinear CV-MV relation where T_{oSS} the outlet temperature at steady state.

The time-dependent model for the HX process is assumed to follow first-order linear dynamics represented by a Hammerstein model. The first-order linear dynamics for Equation (3-13) is developed which is shown in Equation (3-14).

$$\tau \frac{dT_o}{dt} + T_o = T_{oSS_{\,t-\theta}} = T_{i_{t-\theta}} + \frac{C}{F_{w_{t-\theta}}} X_{s_{t-\theta}}^{\,b} \tag{3-14}$$

In Equation (3-14), τ represents the time constant, (t-θ) represents the delay in time. The constants are lumped together as parameter C represented by Equation (3-15).

$$C = \frac{\lambda a}{\rho C_p} \tag{3-15}$$

The model developed is a simple regression model. In order to find the model parameters for regression modeling the process data are required. To capture the behavior of the process across a wide range of operating conditions, a skyline function is established by randomly changing the water flow rate, steam valve opening (MV) and the hold times for the water flow rate and the steam valve opening.

For generating the skyline function, the water flow rate is allowed to randomly move from 1 to 11 gal/min. The steam valve is allowed to randomly move from 5 to 80%. Both the ranges are the minimum and maximum operating range of the HX001. It was taken into consideration that the hold time should neither be too less, for the process not to be able to reach steady state nor too long for the process to be at steady state for a long time. Hence the hold times are randomly set to values between 5 and 270 seconds for both the water flow rate and the steam valve position. The maximum of the hold time range is computed as the sum of the delay in the process and thrice the time constant of the process. The approximate delay in the process is estimated visually. The outlet temperature of water is recorded by changing the water flow rate, steam valve opening and the hold times. The skyline data is thus generated.

Original Leapfrogging and Leapfrogging with improved initialization is used to fit the experimental data with a dynamic model to find the best set of parameters. The model parameters obtained by using the Leapfrogging and its improvement are τ, θ, b and C. The results obtained by implementing original Leapfrogging is compared with those obtained through improved initialization method. The applicability of the improved initialization on a 4 DV problem is demonstrated through this modeling.

3.3 Controller Development:

The following section discusses the methodology for Aim 3: Implement constrained nonlinear horizontal predictive control on the HX and use original and improved Leapfrogging to solve for the future manipulated variable (MV) moves:

In order to establish credibility, nonlinear HPC is established on the nonlinear pilot scale HX equipment for the CV-MV relation. As discussed in Chapter 2 (Background), most of the industrial applications use broadly accepted linear models. In this case since the process is nonlinear, a nonlinear horizon predictive control is used for the control action.

Figure 3-11 represents the block diagram of the HPC controller implemented on the HX process. In Figure 3-11, T_{SP} represents the set point temperature of the cold side water, pmm represents the process model mismatch, T_{SP}' represents the biased set point, r represents the reference trajectory, U represents the control action input, P the HX process and M the first order dynamic model, T_M represents the modeled temperature and T_P represents the process temperature. The objective of the controller to is make the process temperature (T_P) reach the set point temperature value (T_{SP}).

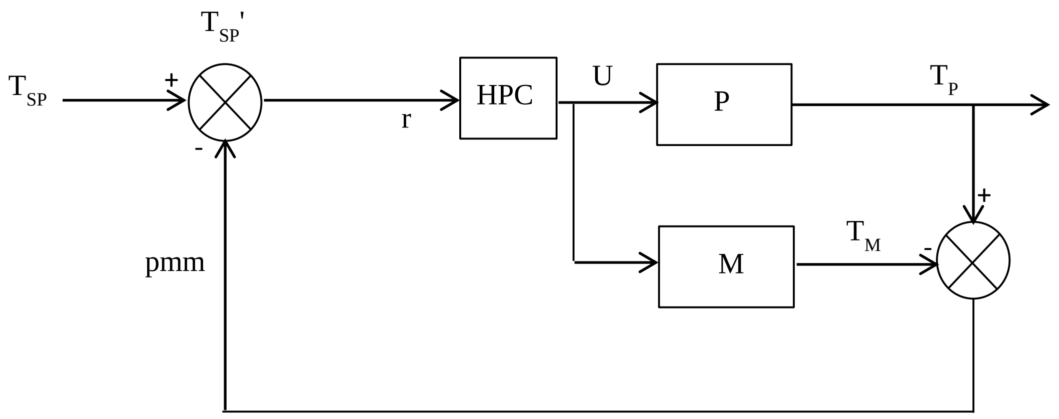


Figure 3-11: HPC Controller Implementation Block Diagram

The entire HPC controller is built on the LabView interface shown in Figure 3-10 including both the MANUAL and the AUTO modes.

3.3.1 MANUAL Mode:

MANUAL mode represents the open loop system wherein there is no requirement for the controller to make output decisions. The set point temperature (T_{SP}) is tracking the process temperature (T_P) in order to avoid any bump while switching to AUTO mode. The modeled temperature value (T_M) is computed at every time instant. The inlet temperature (Ti), lumped parameter (C), power law model constant (b), steam valve opening (U), flow rate of water (F_w), sampling interval (dt) and time constant (τ) are given as inputs for the model. Delay is incorporated for the model calculation. The modeled temperature value at every instance uses the model temperature value from the previous instance. There is an override on the steam valve opening which overrides U with 100 if U is greater than 100 and U with 0 if U is less than 0.

Equation (3-16) is the analytical representation of Equation (3-14) where U is same as X_s which represents the steam valve position. This Equation (3-16) is used for computing the modeled temperature value in the P2N model.

$$T_m = T_{i_{t-\theta}} + \left(\frac{C}{F_{w_{t-\theta}}} U_{t-\theta}{}^{b} - T_m\right)\frac{dt}{\tau} + T_m \tag{3-16}$$

$$pmm = T_p - T_m \tag{3-17}$$

Equation (3-17) is used to compute the process model mismatch pmm.

3.3.2 AUTO Mode:

AUTO mode represents the closed loop system where feedback is sent to the controller to make the output decisions or MV moves. The AUTO mode consists of three parts namely the past-to-now model (P2N), the now-to-future model (N2F) and the optimizer (Leapfrogging optimization technique). Similar to MANUAL model, in AUTO mode there is an override on the steam valve opening which overrides U with 100 if U is greater than 100 and U with 0 if U is less than 0. The biased set point and the reference trajectory are computed in the AUTO mode.

The biased set point is computed using Equation (3-18).

$$T_{SP}' = T_{SP} - pmm \tag{3-18}$$

The analytical representation of reference trajectory is shown in Equation (3-19).

$$r = \frac{dt}{period} T_{SP}' + (1 - \frac{dt}{period}) r \tag{3-19}$$

In Equation (3-19), period represents the controller tuning value.

3.3.3 P2N Model:

The P2N model gives the current model value to the optimizer and N2F model to find the best set of control actions to make the process reach the set point. The P2N model calculates the current process value using the control action from the previous time stamp using Equation (3-16). The process model mismatch (pmm) is also computed using the P2N model. The model is computed using Equation (3-17). The inputs for computing the modeled value at the current time instant model is given to the P2N model.

3.3.4 N2F Model:

The N2F model is similar to the P2N model but, N2F model computes the modeled temperature (T_M) for every time stamp over a future horizon using Equation (3-16). N2F model is used to predict the future behavior of the process in order to find the best set of control actions to be implemented. Horizon is the time by which the controller aims at moving the process towards the set point. The modeled temperature of the P2N model (P2NT_m) is given as an input to initialize the N2F model. Since the pmm is already computed in P2N, its given as an input from the P2N model. The biased set point (T_{SP}') is computed using Equation (3-18). The reference trajectory (r) is computed at every time stamp using Equation (3-19). Both the T_{SP}' and r are computed within the sub routine where N2F model is computed. In this study 3 control actions (MV moves) are computed at every time stamp. The control action implementation across the horizon is as follows

1. Until the time delay, the previous control action (Ud) is implemented
2. After the delay, until (delay + 25 % of horizon), the first control action (U1) is implemented
3. From (delay + 25 % of horizon) to (delay + 50 % of horizon), the second control (U2) action is implemented
4. The rest of the time until the end of the horizon, the third control action (U3) is implemented.

The control action as mentioned above is implemented to the N2F model and the T_M values are computed across the horizon at every time stamp.

The reference trajectory is the path that the CV needs to take in order to reach the T_{SP}. Since the model is designed to be the proxy of the process, the optimizer and the N2F model work in tandem to reduce the sum of square deviation (SSD) between the model and the reference trajectory. The SSD across the horizon is computed as described in Equation (3-20) where t represents the time stamp. This SSD is given to the optimizer as OF. In Equation (3-20), r and T_m vary with time.

$$\mathrm{SSD} = \sum_{t=1}^{\mathrm{horizon}} (r - T_m)^2 \tag{3-20}$$

3.3.5 Optimizer:

The objective of the optimizer is to minimize the SSD. The SSD value is computed within the N2F model and is given to the optimizer. The optimizer finds the optimum value of U1, U2 and U3 so that the SSD is minimized as represented by Equation (3-21).

$$\mathrm{OF} = \mathrm{Min}_{\{U1,U2,U3\}}\mathrm{SSD} \tag{3-21}$$

In this study Leapfrogging optimization technique is used as an optimizer for the HPC. Since 3 optimum values have to be computed to minimize the OF, the optimization problem is a 3-DV problem. Hence 30 players are randomly initialized across the feasible OF space. The feasible OF space is the space in which the DV values are between 0 and 100.

At 30 random spots, the optimizer gives the N2F U1, U2 and U3 values and the N2Fgives to the optimizer the corresponding OF values (SSD). The Leapfrogging optimization proceeds by finding the best and the worst OF value and leaping over the worst across the best into the reflected hyper volume until all players converge. The convergence criteria used in this case is represented by Equation (3-22).

$$\frac{(\mathrm{DV1B}-\mathrm{DV1W})^2+(\mathrm{DV2B}-\mathrm{DV2W})^2+(\mathrm{DV3B}-\mathrm{DV3W})^2}{3} < 1\mathrm{E}-5 \tag{3-22}$$

In Equation (3-22), DV1B and DV1W represents the DV coordinate value for the best and the worst player in the DV1 axis, DV2B and DV2W represents the DV coordinate value for the best and the worst player in the DV2 axis and DV3B and DV3W represents the DV coordinate value for the best and the worst player in the DV3 axis.

The stopping criteria is computed as the mean of the distance squared between the best and the worst in every dimension to be less than a tolerance.

The optimizer stops when the stopping criteria is achieved and the first control action, first MV move, U1 is stored at every time stamp. The stored MV moves (U1) are given to the P2N model in a delayed fashion.

Since the optimizer minimizes the SSD between the modeled temperature and reference trajectory at every time stamp, feeding the optimized control action makes the process reach its set point.

3.3.6 Tuning:

The controller is tuned in order to achieve good stability and fast response in terms of making the process reach the set point. A balance between stability and response speed is desired for tuning the controller. Stability of the controller is indicated by the number of times the controller makes the process undershoot and overshoot the set point. Lesser the over and under shoots, better is the stability of the controller.

The tuned value of the controller (period) is used in the computation of reference trajectory (r) using Equation (3-19). Lower the tuning parameter values, higher is the aggressiveness and higher the tuning parameter values is higher is the sluggishness of the controller. The tuning is done in order to strike a balance between the controller being very aggressive and very sluggish. Only the period parameter is changed to adjust the tuning of the controller.

CHAPTER 4

EXPERIMENTATION

The experimentation for each of the specific aims discussed in Chapter 1 (Introduction) is discussed below.

4.1 Improved Initialization Experimentation:

The following section discusses the experimentation for Aim 1: Provide fundamental knowledge and further improve efficiency of the technique [8].

The improved initialization is tested on several functions. Four two-dimensional (2-DV) functions are considered and one min-max application of DVs from 1 to 8 is explored. F1 is the Goldstein-Price function [1, 8] which is an irregularly shaped function with a possible flat valley between steep and walls. Four local minima are present in the valley. Equation (4-1) represents the mathematical equation for function F1.

$$\begin{aligned} \mathrm{OF} = {} & (1+(\mathrm{DV1}+\mathrm{DV2}+1)^2(19-14\mathrm{DV1}+3\mathrm{DV1}^2-14\mathrm{DV2} \\ & +6\mathrm{DV1DV2}+3\mathrm{DV2}^2)(30+(2\mathrm{DV1}-3\mathrm{DV})^2(18-32\mathrm{DV1} \\ & +12\mathrm{DV1}^2+48\mathrm{DV2}-36\mathrm{DV1DV2}+27\mathrm{DV2}^2) \end{aligned} \tag{4-1}$$

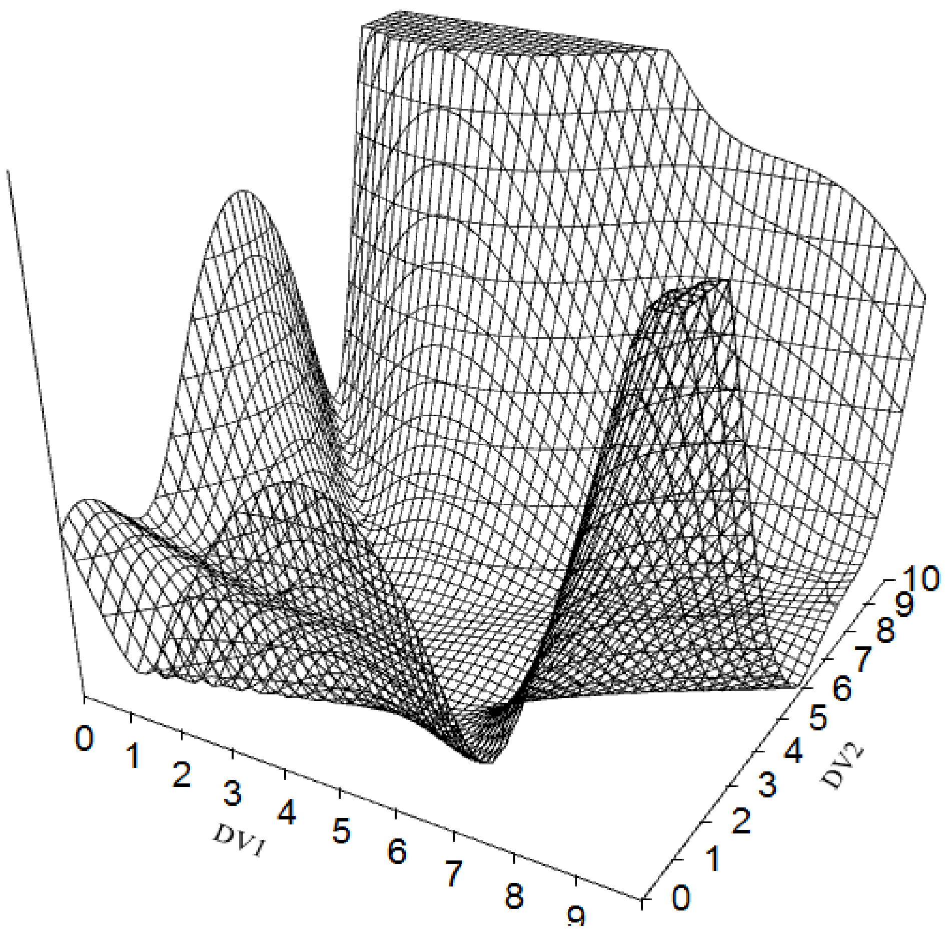


Figure 4-1: 3D View of F1 (Goldstein & Price)

Figure 4-1 shows the pictorial 3D view of the function F1.

F2 represents the Peaks function [1, 8] that has three well-shaped minimum. Equation (4-2) demonstrates the mathematical equation for Function F2.

$$\begin{aligned} \mathrm{OF} = {} & 3(1-\mathrm{DV1})^2\,\mathrm{Exp}(-1*\mathrm{DV1}^2-(\mathrm{DV2}+1)^2) \\ & -10\left(\frac{\mathrm{DV1}}{5}-\mathrm{DV1}^3-\mathrm{DV2}^5\right)\mathrm{Exp}(-1\mathrm{DV1}^2-\mathrm{DV2}^2) \\ & \frac{(\mathrm{Exp}(-1(\mathrm{DV1}+1)^2-\mathrm{DV2}^2))}{3} \end{aligned} \tag{4-2}$$

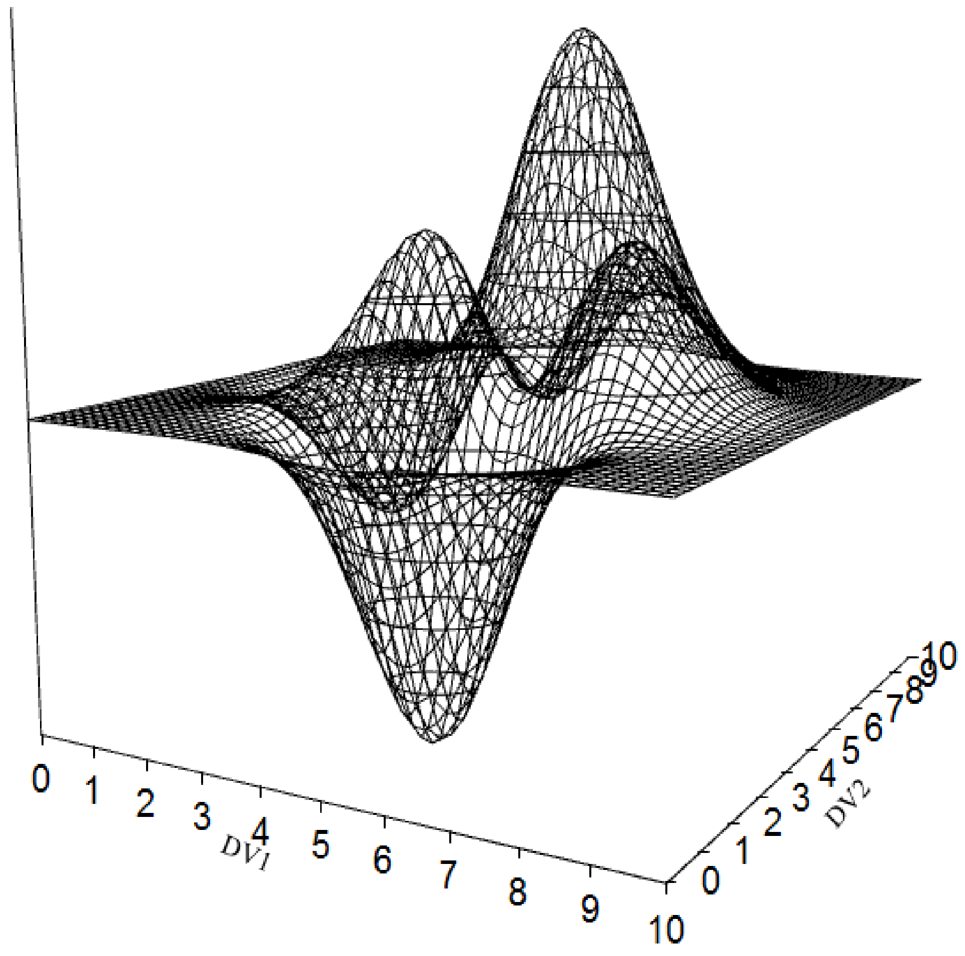


Figure 4-2: 3D View of F2 (Peaks)

Figure 4-2 shows the pictorial 3D view of function F2.

F3 has several more surface aberrations [1, 8]. Sharp Troughs has three minima with a gentle bottom slope. The global optimum is at the proximity of the slope discontinuity in the valley, leading to premature convergence not at the global optimum. Equation (4-3) represents the function F3.

$$OF = (0.02((DV1-8)^2 + (DV2-6)^2 + 15Abs((DV1-2)(DV2-4)) \\ -400Exp(-(DV1-9)^2 + (DV2-9)^2)))(1+(5((DV1-1.5)^2 \tag{4-3}$$

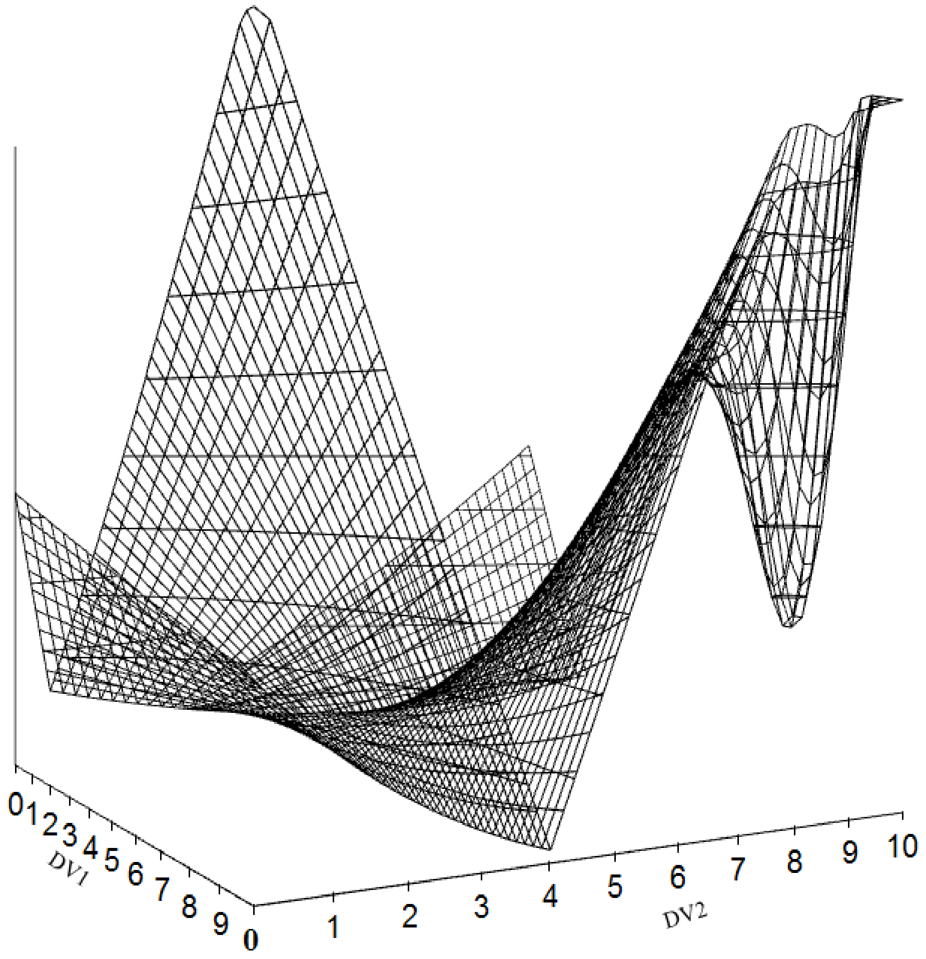


Figure 4-3: 3D View of F3 (Sharp Troughs)

Figure 4-3 shows the pictorial 3D view of function F3.

F4 also has severe features [1, 8]. Function F4 resembles a boot print in the snow with pin hole. Most of the times the solution of an optimization technique gets attracted to the toe of the boot print, although there is a pinhole minimum up on the snow surface. Equation (4-4) represents the function F4.

$$\begin{aligned} OF = (0.5DV1 - 0.2DV2 + 5(\frac{1}{1 + Exp(-3(1 + 0.2(DV2-4)^2 - DV1)}))\\ (1 + (5((DV1-1.5)^2 + (DV2-8.5)^2) - 2)\\ Exp(-4((DV1-1.5)^2 + (DV2-8.5)^2))) \end{aligned} \tag{4-4}$$

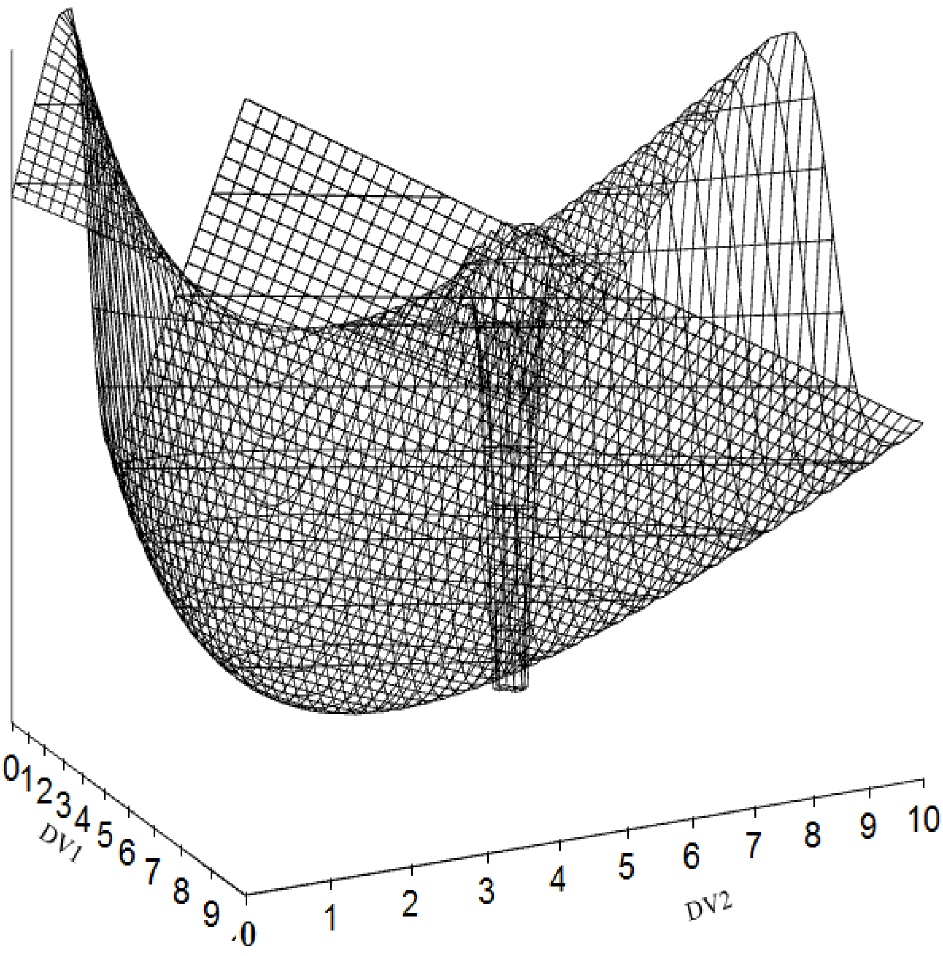


Figure 4-4: 3D View of F4 (Bootprint with Pinhole)

Figure 4-4 shows the pictorial 3D view of function F4.

Equation (4-1) through Equation (4-4) and Figure 4-1 through Figure 4-4 are sourced from the Optimization Applications (ChE 5703) course offered by Dr. R. Russell Rhinehart at OSU in fall 2013.

The Jupitor’s Eye function whose contour is shown in Figure 3-4 and Figure 3-5 is not considered for the experimentation because it is a single optimum function for which the initial surface exploration is not expected to provide any additional benefit as compared to original Leapfrogging optimization technique.

All the 2-DV functions have both the OF and the DVs scaled between 1and10. In order to use the Equation (3-6) for the 2-DV functions, knowledge about f_A is required. For all the 2-DV cases considered the f_A value used is based on the *a priori* knowledge about the surface. The convergence criteria used for the 2-DV cases is based on the DV distance, which is calculated as the root-mean-square deviation from the centroid of the cluster. Leapfrogging stops when the DV distance is <= 0.00001.

The confidence c in Equation (3-6) with which it is said that at least one player during initialization stage is in the Region A is 99.99% in all the 2-DV cases.

The optimization technique is executed many times, so that the results obtained represent broad range of expectations. Executing the optimization technique many times from random starts [47] also avoids the result of one individual fortuitous trial. In this study each of the 2-DV case is executed 10,000 from random initialization. As discussed in Chapter 3 (Methodology), the optimization process consists of two steps:

1. Exploring the surface by placing M_i players randomly
2. Picking the top 20 players and continuing with the Leapfrogging optimization technique

The improved initialization study also includes a high-dimension min-max application.

$$\underbrace{\min J}_{\{x_1, x_2, \ldots x_D\}} = \max(g_1, g_2, \ldots. g_D) \tag{4-5}$$

Where $g_i = \alpha + \beta x_i + \gamma x_i^2 - e^{-\delta(x-\varepsilon)^2}$ such that $0 \le x_i \le 10$

Equation (4-5) represents the higher dimension min-max function. Any value can be given to the coefficients. In this case, the coefficient values used are *α=5, β=1, γ=0.1, δ=1,* and *ε=9.2.* The surface thus obtained is a quadratic surface and it has a broad minimum value (2.5) at x=5. This surface also has a global optimum at the proximity of x=9.25. This min-max function can be enlarged to higher dimensions.

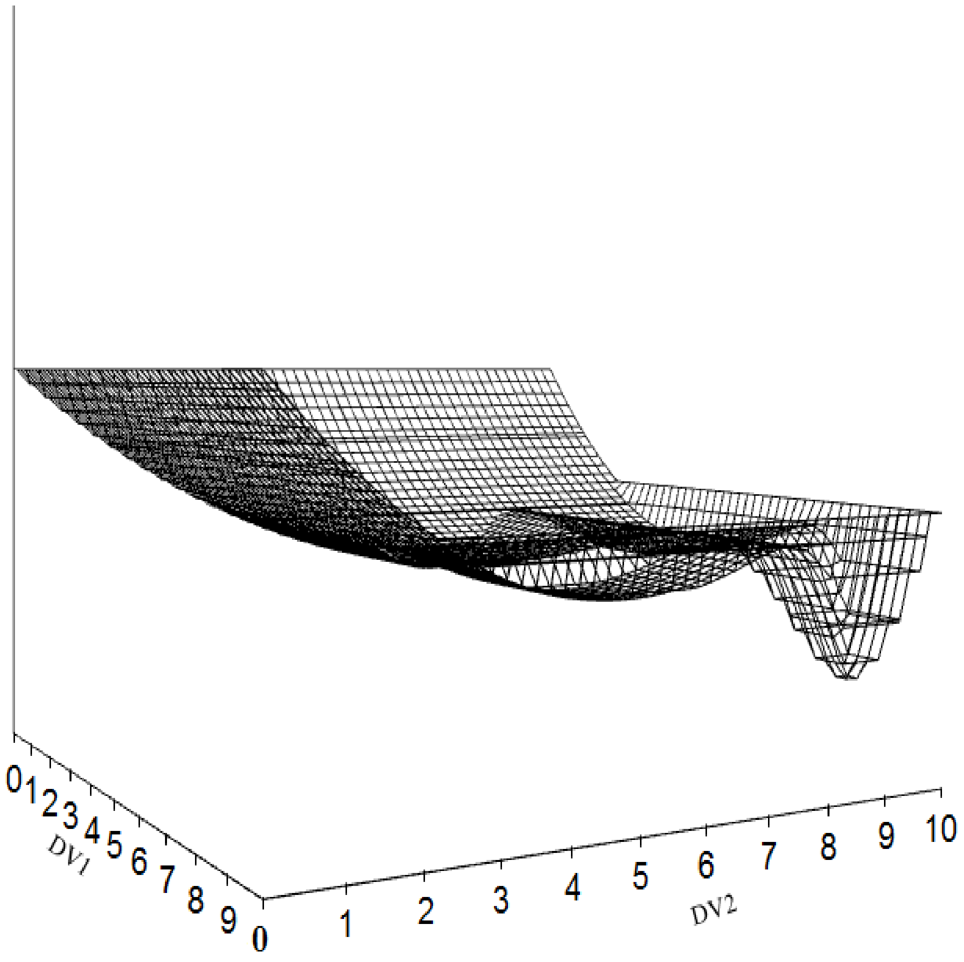


Figure 4-5: 3-D View of 2 DV Min-Max Function

Figure 4-5 shows the pictorial 3-D view of the 2 dimension min max function contour. It is sourced from the Optimization Applications (ChE 5703) course offered by Dr. R. Russell Rhinehart at OSU in fall 2013.

The global attractor area A in this case is considered as 0.156, which represents the fraction of full range in each dimension. Hence, the total global attractor area is given by Equation (4-6).

$$f_A = 0.156^{N_{DV}} \tag{4-6}$$

In Equation (4-6) N_{DV} represents the number of dimensions. For all the min-max cases, the confidence is considered as 99% in order to keep the number of initial players M_i required for initial surface exploration to a reasonable value. The stopping criterion is based on the DV distance which is calculated as the root-mean-square deviation from the centroid of the cluster. Leapfrogging stops if the root mean square deviation (RMS) DV distance is <= 0.001. All cases of the min-max function are executed 10,000 times to obtain results from a broad range of expectation.

In all the above test cases, initially the surface is explored with excessive number of players which is computed based on Equation (3-6). After placing all the players randomly on the surface, the OF value of each player is calculated. The players are sorted based on the OF value and the top 10*DVs players are considered for proceeding with the Leapfrogging optimization technique.

For each case, the number of initial players M_i required for surface exploration is computed using Equation (3-6). ANOFE, CDF and PNOFE is computed for each case using both the Leapfrogging optimization technique and the Leapfrogging with improved initialization technique. ANOFE, CDF and PNOFE in both the cases of the optimization technique is compared against each other. For the original Leapfrogging optimization technique 10*DV players are considered.

All the execution of the Leapfrogging optimization technique and the data analysis is carried out on a Excel-VBA platform.

4.2 HX Modeling Experimentation:

The following section discusses the experimentation for Aim 2: Apply original and improved Leapfrogging on nonlinear pilot scale HX modeling.

As discussed in Chapter 3 (Methodology), a skyline function is generated for the HX process with outlet water temperature as the CV and the steam valve opening as the MV.

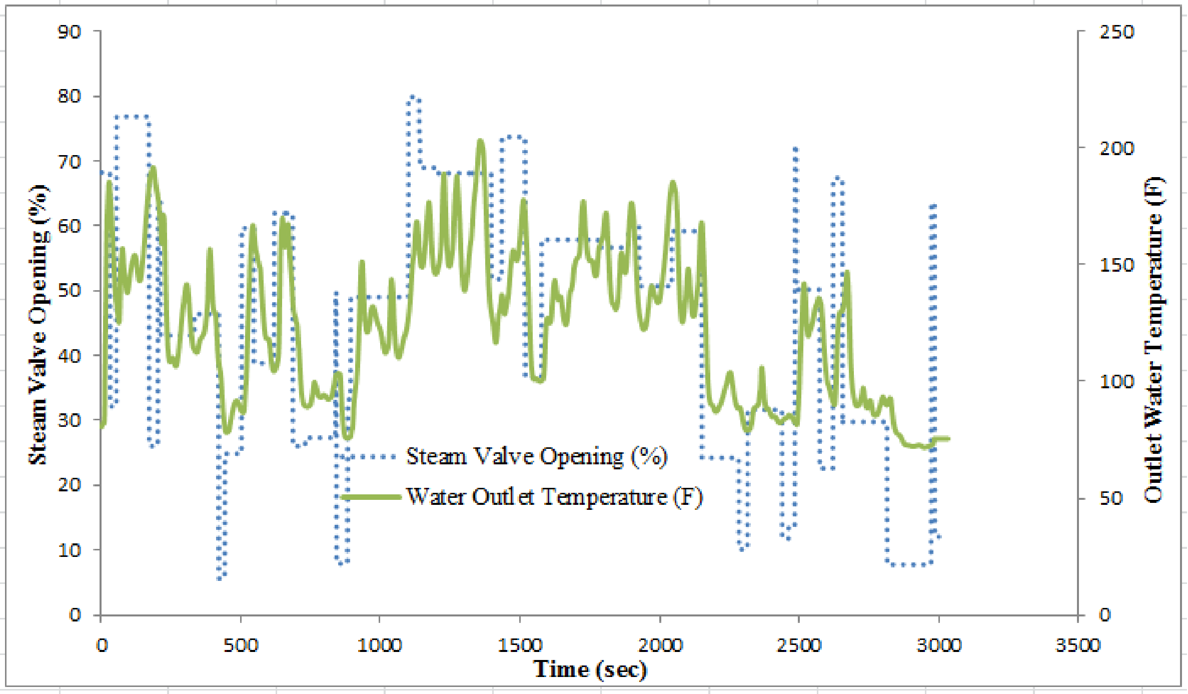


Figure 4-6: Skyline Function with CV-MV Data

Figure 4-6 represents the skyline function generation over a period of about 3000 seconds (50 minutes). The model represented in Equation (3-14) is used to find the best set of model parameters are τ, θ, b and C in order to make the model match the CV data.

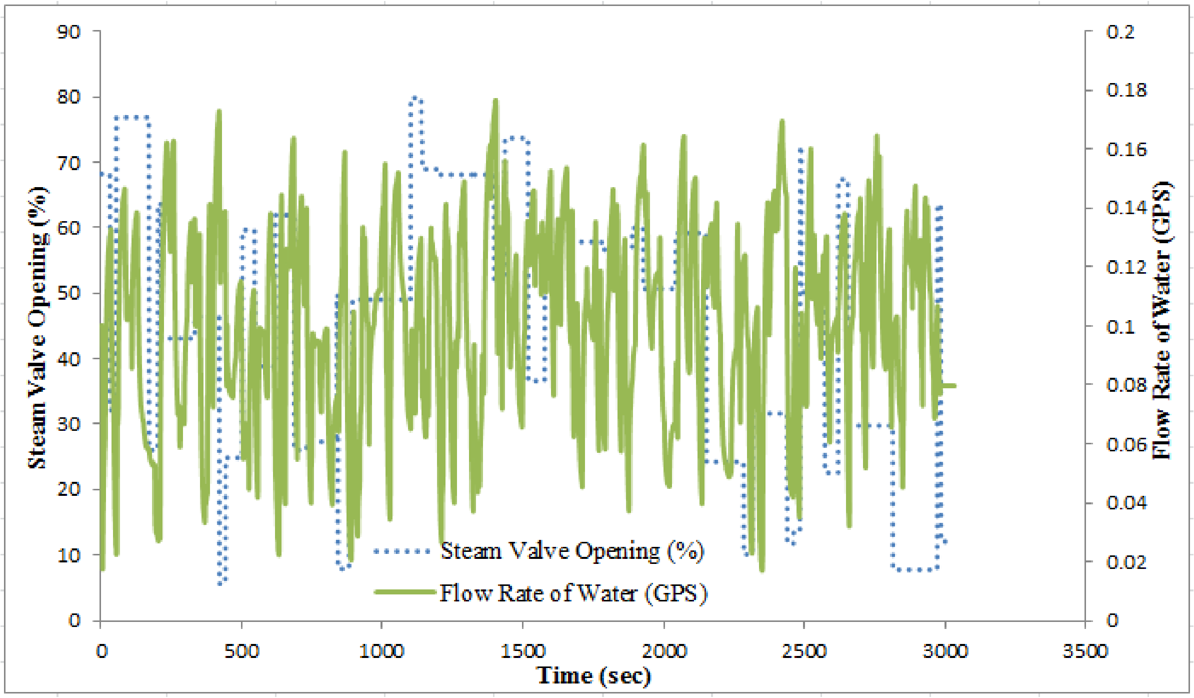


Figure 4-7: Skyline Function with MV and Disturbance Data

The model parameters thus generated are used in the HPC controller. For the controller, the flow rate of water f_w is considered as a disturbance. Figure 4-7 represents the skyline function of the CV value and the disturbance. The disturbance is taken into account while finding the best set of model parameters.

Another set of model parameters is also regressed where the delay is a dynamic delay. In order to make the model better acquire the behavior of the process, the delay of the model is modeled as a function of the flow rate of the water. Instead of modeling and finding the best delay parameter value, the volume of water from the previous time step present still in the HX is modeled. The delay is then computed using Equation (4-7).

$$\theta = \frac{d}{f_w} \tag{4-7}$$

In Equation (4-7), d represents the volume of water from previous time stamp still present in the HX, f_w represents the volumetric flow rate of water and θ represents the dynamic delay in the model. Best set of model parameters τ , d, b and C are regressed using the optimization techniques in order to make the model match the CV data.

Both the original Leapfrogging optimization technique and the Leapfrogging with improved initialization optimization technique are used to find the best set of model parameters. In order to obtain results representing a broad range of expectation, each optimization technique is executed 5,000 times. The optimization techniques are built and executed in the Excel-VBA platform.

Convergence on the HX regression modeling to find the best set of model parameters is determined when the root-mean-square (RMS) deviation between the CV data and the model from a random subset consisting of 25 % of the skyline function data indicates no improvement statistically in the model data relative to making the model further match the CV data [1]. The execution of the optimization technique is stopped when the RMS value is less than 0.8 [1].

For both the optimization techniques after 5000 trials PNOFE value is computed using Equation (2-3). PNOFE scale is used to compare the original Leapfrogging and the Leapfrogging with improved initialization technique against each other.

4.3 HPC Controller Experimentation [54]:

The following section discusses the experimentation for Aim 3: Implement constrained nonlinear horizontal predictive control on the HX and use original and improved Leapfrogging to solve for the future manipulated variable (MV) moves:

Several experiments are conducted to test the working of the HPC controller. Both static and dynamic delay models are used in the controller representing two cases. The experiments are designed to demonstrate the robustness and accuracy of the controller in both the cases. The experiments are listed below:

4.3.1 Bumpless Transfer:

In order to avoid any type of process upset, the controller when transferred from MANUAL mode to AUTO mode should be bumpless. The HPC controller is initially run in MANUAL mode until the CV reaches steady state and is transferred to AUTO mode to test if the transfer is bumpless and the set point is retained by the controller.

4.3.2 Disturbance Rejection:

The HPC controller is built based on a feedback mechanism. At every time stamp the output from the previous time stamp has an impact on the inputs to the current time stamp. Any disturbance should be rejected by the controller by not allowing it to cause any sort of deviation of the process from the desired set point. The flow rate of the water is changed to test the disturbance rejection capability of the controller.

4.3.3 Constraint Recovery:

If a physically unrealizable set point is given to the controller (for example too low a set point temperature) which is present in the constrained region of the process (for instance needing the valve to be less than 5 % open), then the process will reach its closest feasible value to the set point. At steady state there will be an offset between the process and the set point because of the unrealizable set point. When the set point is changed to a realizable feasible value, the controller should immediately respond to the realizable set point and not have wind up. If the controller accumulates the error caused due to the unrealized set point and does not immediately respond to the realizable set point, then the controller is said to have wind up.

For conducting this experiment, an unrealizable set point of 70 F is given to the controller. Once the process reaches steady state, the set point is changed to 100 F to test the constraint recovery of the controller.

4.3.4 Rate of Change Constraint Test:

The rate of change of the MV at every time stamp should follow the rate of change constraint. If the rate of change constraint is 5%, two consecutive MV moves should not be more than 5%. If the rate of change constraint is 100%, then two consecutive MV moves should not be more than 100%.

To test this on the controller, initially the rate of change constraint is set to 100% and the MV moves are observed. Once the process reaches steady state the rate of change constraint is then change to 5% and the MV moves are observed.

4.3.5 Controller Tuning:

The behavior of the controller for various tuning parameter value is tested. If the tuning parameter is low, then the reference trajectory moves faster towards to the biased set point making the controller aggressive. If the tuning parameter is high, then the reference trajectory takes a longer time to reach the biased set point making the controller sluggish. This behavior of the controller is tested for different tuning parameter values.

CHAPTER 5

RESULTS AND DISUCSSION

The results for each of the specific aim discussed in Chapter 1 (Introduction) are discussed below.

5.1 Improved Initialization Results:

The following section discusses the results for Aim 1: Provide fundamental knowledge and further improve efficiency of the technique [8].

Table 5-1 represents the number of initial players considered for each of test functions based on Equation (3-6). The global attractor region, A, represented by f_A in Table 5-1 is estimated as an from the function contours because of the *a priori* knowledge about the OF surface in each of the case. The global attractor region in a dimension is estimated from the *a priori* knowledge about the OF surface and then f_A for the entire OF surface is estimated using Equation (5-1).

$$f_A = {f_{A^*}}^{N_{DV}} \qquad (5\text{-}1)$$

In Equation (5-1), f_{A^*} is the global attractor region in a dimension and N_{DV} is the total number of dimensions of the optimization problem.

In Equation Table 5-1 f_{A*} is the global attractor region in a dimension and N_{DV} is the total number of dimensions of the optimization problem The confidence c is the desired probability that after during the initialization stage one player will be in Region A. In all the 2-DV cases the confidence is 99.99% because of the better *a priori* knowledge of the surface and on the rest is 99% because of the usage of Equation (5-1).

Table 5-1: Initial Players for Surface Exploration

Function	c (%)	f_A (%)	M_i (players)
F1	99.99	1.0	916
F2	99.99	4.0	226
F3	99.99	4.88	184
F4	99.99	0.3	3066
min-max-1	99.00	0.1555	27
min-max-2	99.00	0.0242	188
min-max-3	99.00	3.76E-3	1,222
min-max-4	99.00	5.85E-4	7,869
min-max-5	99.00	9.10E-5	50,611
min-max-6	99.00	1.42E-5	325,433
min-max-7	99.00	2.20E-6	2,092,508
min-max-8	99.00	3.42E-7	13,454,598

Table 5-2 compares the ANOFE (average number of function evaluation) for each function with original Leapfrogging and Leapfrogging with improved initialization. Every computation of the OF function accounts as a function evaluation. ANOFE is the average of the number of times the OF is computed across all the trials. The number of times OF value is computed during surface exploration is also accounted for in the ANOFE. Since more players are used by Leapfrogging with improved initialization technique for surface exploration, the ANOFE is higher as compared with that of the original Leapfrogging.

After initialization with M_i players listed in Table 5-2, 10 * Number of DV players are used for the Leapfrogging optimization technique in each case.

Table 5-2: Comparison of ANOFE of Original and Improved Leapfrogging

Function	Global OF Value	Original LF ANOFE	LF with Improved Initialization ANOFE
F1	0.0447	234	1073
F2	0.1326	219	399
F3	0.0800	358	424
F4	-7.2521	289	3225
min-max-1	1.2474	76	134
min-max-2	1.2474	163	403
min-max-3	1.2474	234	1,506
min-max-4	1.2474	299	8,394
min-max-5	1.2474	462	51,191
min-max-6	1.2474	615	326,180
min-max-7	1.2474	792	2,093,300
min-max-8	1.2474	982	13,455,580

Table 5-3 represents the PNOFE values computed using Equation (2-3). The number of independent N trials is computed using Equation (2-2). CDF column represents the percentage times the optimizer converged at a point which is at the vicinity of the global optimum. The confidence c in each test cases to be able to tell that the optimizer converged at the vicinity of the global optimum is considered as the same confidence used for computing the initial number of players required.

In all the test function cases the players are initialized randomly between 0 to 10. On all the 2-DV test cases the optimizer stops if ΔDV <= 0.00001 and in all the min-max function cases the optimizer stops if ΔDV <= 0.001.

Table 5-3: Comparison of PNOFE of Original and Improved Leapfrogging

Function	Global OF value	Original LF		LF with Improved Initialization	
		CDF (%)	PNOFE	CDF (%)	PNOFE
F1	0.0447	79.27	1369	100.00	1073
F2	0.1326	79.23	1283	100.00	399
F3	0.0800	97.78	865	99.99	718
F4	-8.7304	11.97	20869	100.00	3225
min-max-1	1.2474	90.20	149	99.90	89
min-max-2	1.2474	52.40	1,011	99.50	350
min-max-3	1.2474	18.10	5,535	99.20	1,506
min-max-4	1.2474	4.10	32,781	99.00	8,394
min-max-5	1.2474	1.00	211,693	99.00	51,191
min-max-6	1.2474	0.20	1,412,373	99.00	326,180
min-max-7	1.2474	0.05	7,292,766	99.00	2,093,300
min-max-8	1.2474	0.02	22,609,124	99.00	13,455,580

Table 5-3 compares the PNOFE values of the original Leapfrogging algorithm (with random initial start) with the Leapfrogging optimization technique with improved initializations at the beginning that leads to surface explorations. In cases with 100 % as CDF to be able to plug into the Equation (2-2) without computing ln(0), the same value of c is used for f which makes the Equation (2-2) go to a value of 1. PNOFE represents the effort to benefit assessment metric of the optimization technique. Both the columns labeled PNOFE of Table 5-3 list the PNOFE values of the original Leapfrogging and the Leapfrogging with improved initialization. For the improved initialization for each test function initially large number of players were used and the best 10*DV players were selected for proceeding with the Leapfrogging optimization technique. The initial number of players used in each case is specified in Table 5-3.

The results for the 2-DV test functions are based on 10,000 trials while for min-max function, it is based on 5,000 trials. In all the 2-DV cases replicate study using 10,000 trials provided a PNOFE which is sufficiently consistent in all the min-max functions 5,000 trials provided a PNOFE that is sufficiently consistent.

The comparison shown in Table 5-3 is done to study the effect of surface exploration prior to finding the solution using the Leapfrogging optimization technique. It can be seen from Table 5-2 that though a large number of players for surface exploration leads to more function evaluations, the improved initialization leads to a considerable reduction in the PNOFE value (effort to benefit ratio) as compared to that of the ordinary Leapfrogging technique because of high CDF value as shown in Table 5-3. The quantum of reduction in PNOFE in each of the test cases is different because of different surface irregularities associated with each of the test function.

In Table 5-3 in some cases it can be seen that the CDF is more than the initial 99.99% or 99% confidence considered initially. The increment in the CDF values from 99.99% to 100% for the 2-DV test functions and from 99% on the min-max functions are because of the fortuitous placement of players in the Region A due to the Leapfrogging optimization. The theoretical analysis claims that $c \leq$ CDF because of fortuitous placement of players in Region A due to leap overs which are not initially in the Region A. In a situation of no player in the Region A, during a leap over a worst player may leap over into the Region A further leading all the other players to converge at the vicinity of the global optimum.

It can be seen from Table 5-3, that improved initialization reduces the PNOFE in all cases. This demonstrates that the improvement based on the fundamental knowledge led to reduction in the effort to benefit computation of the optimization technique.

The improved initialization is helpful only if multiple optima is present with one as the global and rest as the local. If there is presence of only one single optimum then performing the initial surface exploration increases the number of function evaluation, ANOFE for no considerable increase in the benefit (increase in CDF leading to reduction of PNOFE).

In order to determine the initial number of players required for the surface exploration, f_A is required. A perfect knowledge about the Region A is not required. An approximate estimate appears to be sufficient to compute the initial number of players required for surface exploration.

From the results and discussion above following is summarized:

1. Based on fundamental analysis and support theory the claim is that $c \leq$ CDF which means the actual probability of converging at the vicinity of the global is greater than the confidence with which it was expected to converge due to the presence of at least 1 player in Region A during initialization.
2. From a practical point of view, the hope is that the improved initiation will reduce PNOFE because of the increase in CDF value with improved initialization as compared to the original Leapfrogging technique. But if the number of players required for surface exploration leads to higher function evaluations than the benefit caused by increased CDF then, the PNOFE is not going to be lower than that for original Leapfrogging.
3. The data from the experiments discussed above supports the theoretical analysis but does not prove the hope.

5.2 HX Modeling Experimentation Results:

The following section discusses the results for Aim 2: Apply original and improved Leapfrogging on nonlinear pilot scale HX modeling.

5.2.1 Regression Modeling With Static Delay:

Regression modeling of the HX process data obtained with static delay is done using both original Leapfrogging and Leapfrogging with improved initialization. The players for both original Leapfrogging and Leapfrogging with improved initialization are initialized across the same range. Table 5-4 shows the initialization range for the parameters used. The ranges for the parameters are decided based on the process knowledge and the behavior of the process.

Table 5-4: Initialization Range for Parameters (Static Delay)

Parameter	Initialization Range
C	0 - 1
b	0 - 3
τ	10 – 100 s
θ	0 – 11s

Since the regression modeling needs best values for 4 parameters, this is a 4-DV problem and so 40 players are used for the Leapfrogging optimization. In the case of Leapfrogging with improved initialization, 23025 players are used for initial surface exploration and the top 40 players are used for proceeding with the Leapfrogging optimization technique. For computing the number of players required for initial surface exploration using Equation (3-6), f_A is considered as 0.0001 and c is considered as 0.90. Since there is no *a priori* knowledge about the OF surface in this case, a safe bet of 0.1 for the Region A in every dimension is considered. Hence in all 4 dimensions, overall, the Region A is 4th power of 0.1 and so 0.0001 is considered. Since, the

Region A computation is approximate a 90% confidence is assumed. Plugging the values of c and f_A in Equation (3-6) shows the need for 23025 initial players. After the intial surface exploration top 40 players are used for Leapfrogging optimization.

Leapfrogging and Leapfrogging with improved initialization are run 5000 times because the PNOFE obtained in multiple sets of 5000 trials showed to be consistent within 5 % tolerance.

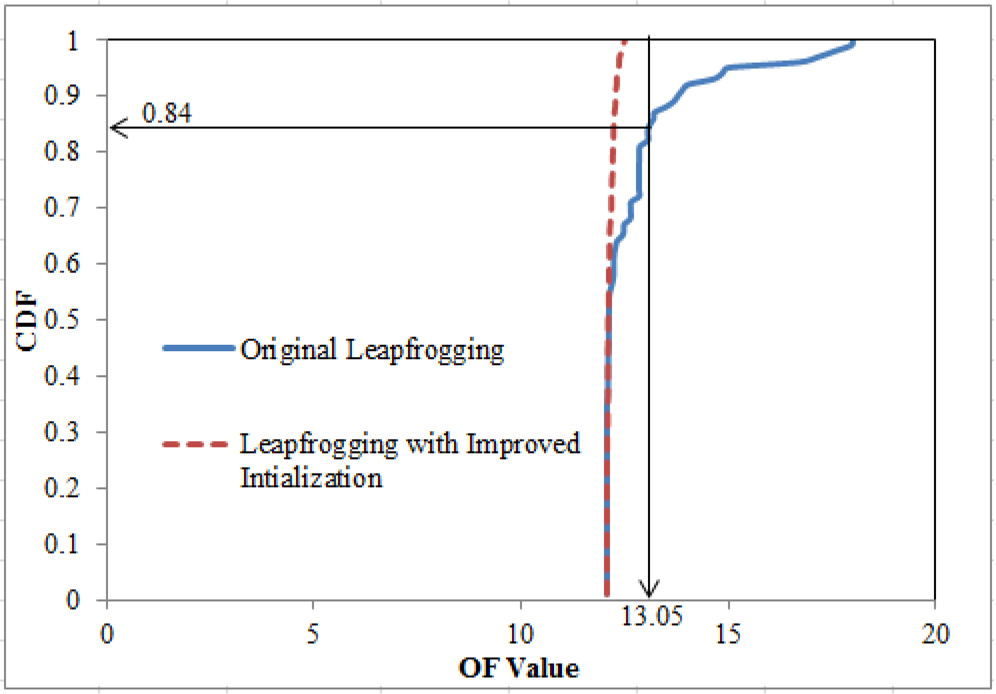


Figure 5-1: CDF vs OF for Regression Modeling With Static Delay

Figure 5-1 shows the CDF for the regression model of the HX modeling using Leapfrogging and Leapfrogging with improved initialization. The dotted line represents the CDF vs OF trend for Leapfrogging with improved initialization while the solid line shows the CDF vs OF trend for the original Leapfrogging optimization technique. Any OF value less than 13.05 is considered as being at the vicinity of the global optimum. The solid line shows that 84 % of the times Leapfrogging converged at the vicinity of the global. The dotted line shows that 100 % of the times the Leapfrogging with improved initialization converged at the vicinity of the global optimum.

Table 5-5 compares the original Leapfrogging with improved initialization Leapfrogging. The confidence c for computing PNOFE is considered as 90 % for both the optimization cases. Since the CDF of 1 cannot be considered for plugging in the PNOFE computation due to high probability of converging at the vicinity of the global 99.99 % is considered for the CDF value. Increasing the number of players did help in achieving a higher probability of attaining the global. As per the theoretical analysis for improved initialization, the CDF obtained is greater than the confidence with which the initial number of players required is estimated. But the increase in the number of initial players caused higher function evaluations which are more than the benefit caused due to the increase in the CDF value, and so, the PNOFE for Leapfrogging with improved initialization is greater than the PNOFE for original Leapfrogging.

Table 5-5: Original and Improved Leapfrogging Comparison (Static Delay)

Parameter	Original Leapfrogging	Leapfrogging with Improved Initialization
C	0.03	0.03
b	1.37	1.37
τ	24.52 s	24.52 s
θ	9 s	9 s
ANOFE	320	23282
CDF	0.84	1.00
PNOFE	403	5820

Figure 5-2 shows the regressed model prediction of outlet water temperature response. The dotted line represents the actual CV response obtained experimentally. The solid line represents the modelled prediction with the best set of model parameters. As can be seen in Figure 5-2 the modeled prediction does not match the experimental data perfectly but it has captured the trend of the CV response. For implementing HPC, the set of model parameters that capture the CV response trend is sufficient.

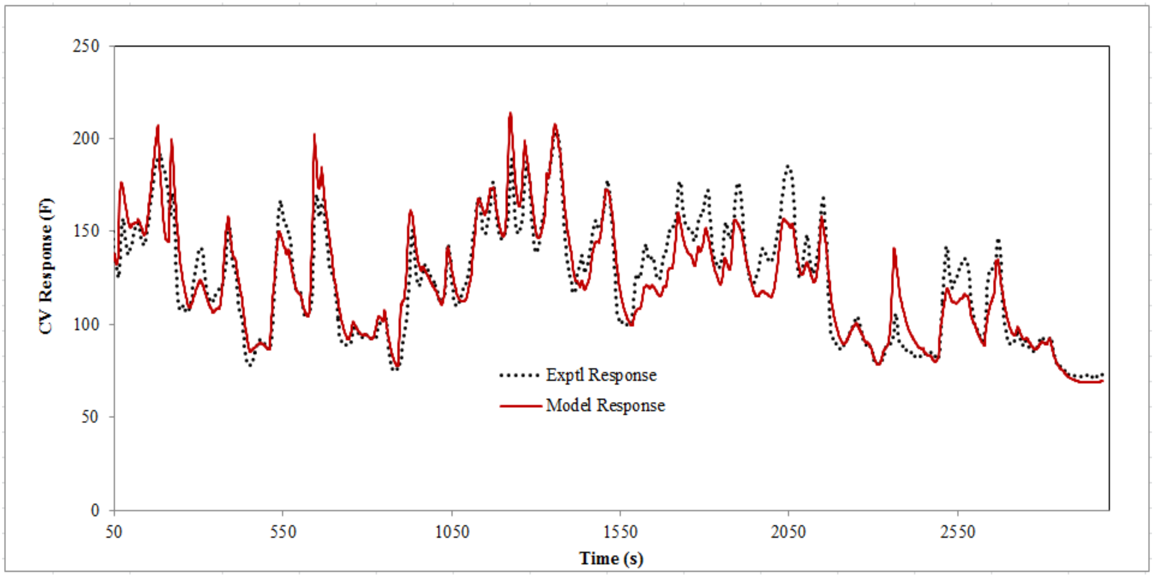


Figure 5-2: Best Model With Static Delay

5.2.2 Regression Modeling With Dynamic Delay:

Regression modeling with dynamic delay is performed using both Leapfrogging and Leapfrogging with improved initialization optimization techniques. Table 5-6 is used for as the initialization range for randomly initializing the players for both Leapfrogging and Leapfrogging with improved initialization.

Table 5-6: Initialization Range for Parameters (Dynamic Delay)

Parameter	Initialization Range
C	0 - 1
b	0 - 3
τ	10 – 100 s
d	0 – 1

As discussed in Section 5.2.1, the initial number of players considered for surface evaluation is

23025 in this case as well. After surface exploration 40 best players are used for continuing with the Leapfrogging optimization technique.

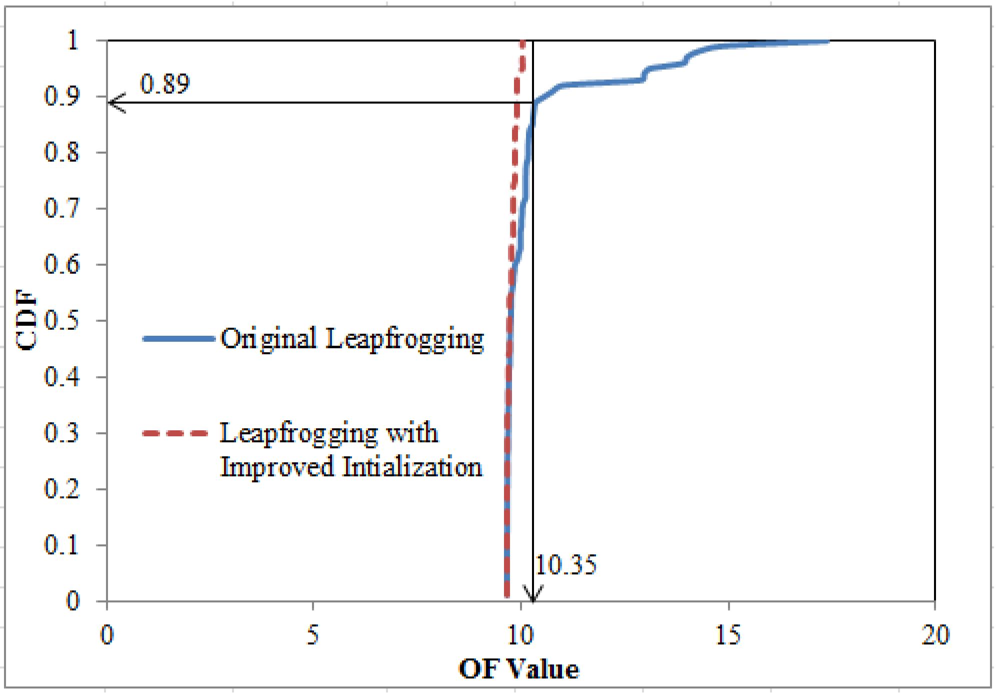


Figure 5-3: CDF Vs OF for Regression Modeling With Dynamic Delay

Figure 5-3 shows the CDF for the regression model of the HX modeling using Leapfrogging and Leapfrogging with improved initialization. The dotted line represents the CDF vs OF trend for Leapfrogging with improved initialization while the solid line shows the CDF vs OF trend for the original Leapfrogging optimization technique. Any OF value less than 10.35 is considered as being at the vicinity of the global optimum. The OF value in the case of dynamic delay is better than the OF value in the case of static delay. The solid line shows that 89 % of the times Leapfrogging converged at the vicinity of the global. The dotted line shows that 100 % of the times the Leapfrogging with improved initialization converged at the vicinity of the global optimum.

Table 5-7 compares the original Leapfrogging with improved initialization Leapfrogging. The confidence c for computing PNOFE is considered as 90 % for both the optimization cases. Since the CDF of 1 cannot be considered for plugging in the PNOFE computation due to high probability of converging at the vicinity of the global 99.99 % is considered for the CDF value of Leapfrogging with improved initialization. As per the theoretical analysis for improved initialization, the CDF obtained is greater than the confidence with which the initial number of players required is estimated. Increasing the number of players did help in achieving a higher probability of attaining the global. But the increase in the number of initial players caused higher function evaluations which are more than the benefit caused due to the increase in the CDF value, and so, the PNOFE for Leapfrogging with improved initialization is greater than the PNOFE for original Leapfrogging.

Table 5-7: Original and Improved Leapfrogging Comparison (Dynamic Delay)

Parameter	Original Leapfrogging	Leapfrogging with Improved Initialization
C	0.05	0.05
b	1.25	1.25
τ	22.88 s	22.88 s
d	0.58	0.58
ANOFE	310	23265
CDF	0.89	1.00
PNOFE	324	5817

Figure 5-4 shows the regressed model prediction of outlet water temperature response. The dotted line represents the actual CV response obtained experimentally. The solid line represents the modelled prediction with the best set of model parameters. As can be seen in Figure 5-4 the modeled prediction does not match the experimental data perfectly but it has captured the trend of the CV response. For implementing HPC, the set of model parameters that capture the CV response trend is sufficient.

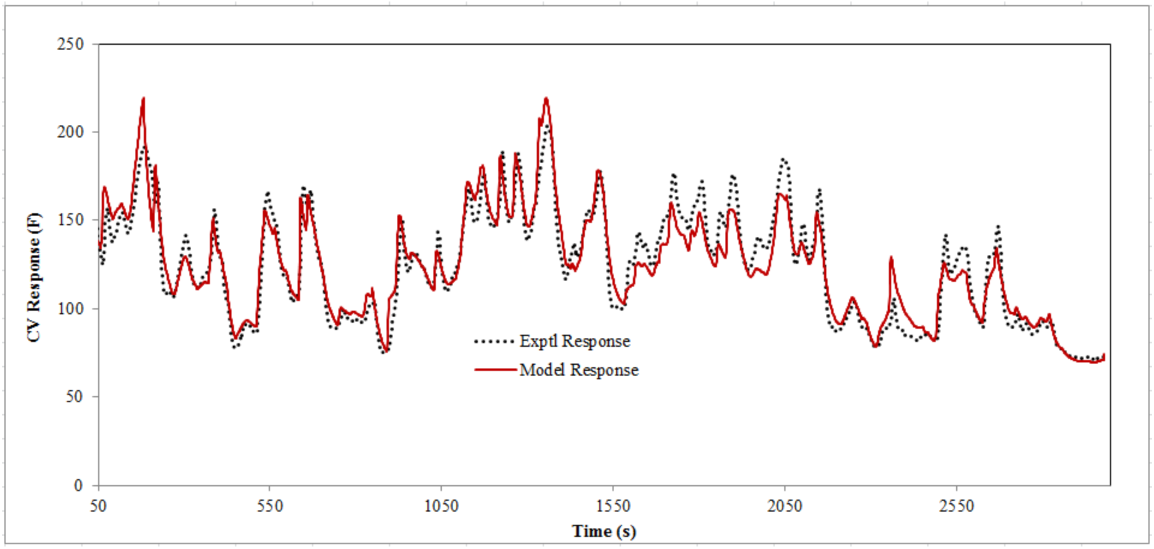
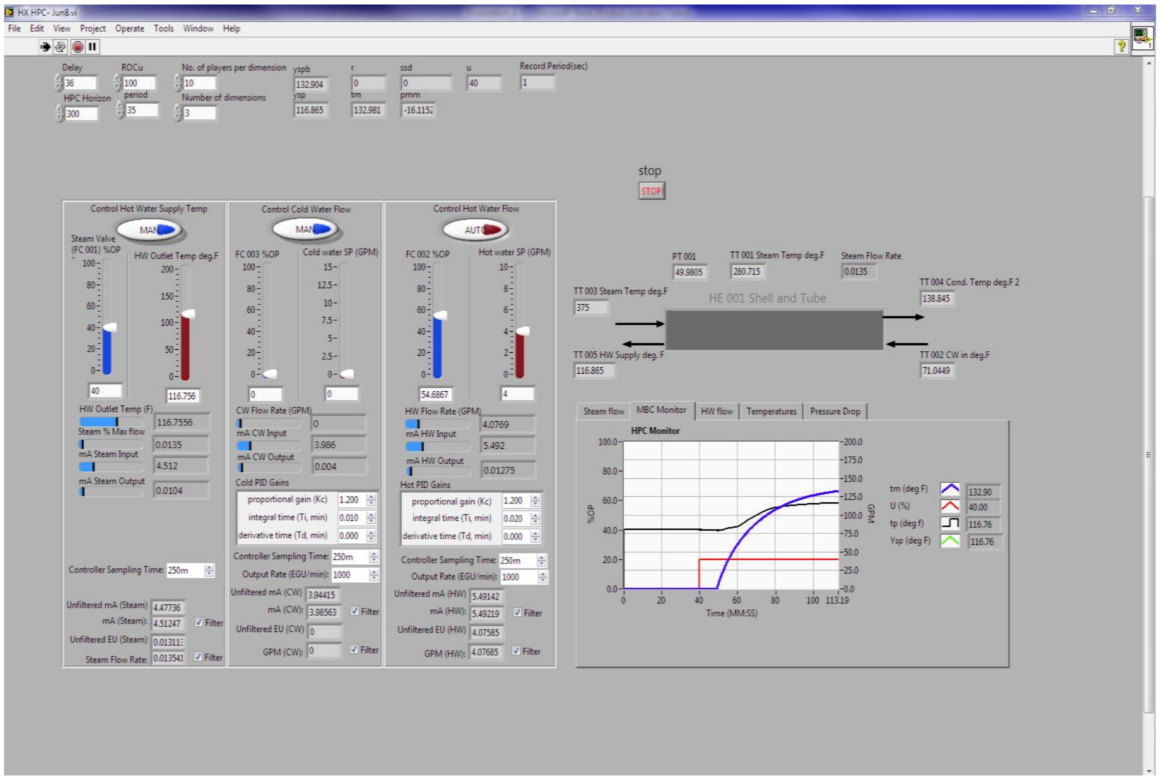


Figure 5-4: Best Model With Dynamic Delay

5.3 HPC Controller Results:

The following section discusses the results for Aim 3: Implement constrained nonlinear horizontal predictive control on the HX and use original and improved Leapfrogging to solve for the future manipulated variable (MV) moves:

Figure 5-5 is the HPC interface built on a LabVIEW platform in UOL at OSU. The HPC control program is built on the already existing LabView program for HX 001 which uses regulatory controls in the UOL. The interface shown in Figure 5-5 is a modification of the previously existing interface.


HX HPC- Jun8.vi

File Edit View Project Operate Tools Window Help

Delay: 36
HPC Horizon: 300
ROCu: 100
period: 35
No. of players per dimension: 10
Number of dimensions: 3
yspb: 132.904
ysp: 116.865
r: 0
tm: 132.981
ssd: 0
pmm: -16.1152
u: 40
Record Period(sec): 1

stop: STOP

Control Hot Water Supply Temp: MAN
Steam Valve (FC 001) %OP: 40
HW Outlet Temp deg.F: 116.756
HW Outlet Temp (F): 116.7556
Steam % Max flow: 0.0135
mA Steam Input: 4.512
mA Steam Output: 0.0104
Controller Sampling Time: 250m
Unfiltered mA (Steam): 4.47736
mA (Steam): 4.51247 [✓ Filter]
Unfiltered EU (Steam): 0.013113
Steam Flow Rate: 0.013541 [✓ Filter]

Control Cold Water Flow: MAN
FC 003 %OP: 0
Cold water SP (GPM): 0
CW Flow Rate (GPM): 0
mA CW Input: 3.986
mA CW Output: 0.004
Cold PID Gains
proportional gain (Kc): 1.200
integral time (Ti, min): 0.010
derivative time (Td, min): 0.000
Controller Sampling Time: 250m
Output Rate (EGU/min): 1000
Unfiltered mA (CW): 3.94415
mA (CW): 3.98563 [✓ Filter]
Unfiltered EU (CW): 0
GPM (CW): 0 [✓ Filter]

Control Hot Water Flow: AUTO
FC 002 %OP: 54.6867
Hot water SP (GPM): 4
HW Flow Rate (GPM): 4.0769
mA HW Input: 5.492
mA HW Output: 0.01275
Hot PID Gains
proportional gain (Kc): 1.200
integral time (Ti, min): 0.020
derivative time (Td, min): 0.000
Controller Sampling Time: 250m
Output Rate (EGU/min): 1000
Unfiltered mA (HW): 5.49142
mA (HW): 5.49219 [✓ Filter]
Unfiltered EU (HW): 4.07585
GPM (HW): 4.07685 [✓ Filter]

HE 001 Shell and Tube
PT 001: 49.9805
TT 001 Steam Temp deg.F: 280.715
Steam Flow Rate: 0.0135
TT 003 Steam Temp deg.F: 375
TT 004 Cond. Temp deg.F 2: 138.845
TT 005 HW Supply deg. F: 116.865
TT 002 CW in deg.F: 71.0449

Steam flow | MBC Monitor | HW flow | Temperatures | Pressure Drop

HPC Monitor
tm (deg F): 132.90
U (%): 40.00
tp (deg f): 116.76
Ysp (deg F): 116.76

Figure 5-5: HPC Controller Interface

In Figure 5-5, the Control Hot Water Supply Temp block is where the outlet water temperature set point is given. The slider on the right is for the set point. The slider on the left indicates the steam valve opening percentage. The Control Hot Water Flow block is where the flow rate of water is specified. The slider on the right is used for specifying the required flow rate of water and the slider on the left adjusts the valve opening to allow the required flow rate of water to pass through. In both the blocks the switch present at the top helps shift between AUTO and MANUAL modes. The 6 boxes with white background present on the interface is where the user inputs delay, horizon, rate of change constraint, tuning parameter, number of players per dimension and number of dimensions are specified. The remaining 8 boxes with a gray background present at the top of the interface adjacent to the 6 boxes displays the biased set point, set point, reference trajectory, model temperature, sum of square deviation, process model mismatch and steam valve position values while the controller is running at every time instant.

The record period shows the data recording time interval. The graph on the bottom right shows the HPC controller behavior on a continuous basis.

The controller routine is executed once every second. In both the cases of HPC controller with static and dynamic delay, root mean square deviation between the worst and best players in each dimension is computed for stopping the optimization technique. If the root mean square deviation is less than 0.00001 then the optimizer stops.

5.3.1 HPC Controller With Static Delay:

Table 5-8 lists the values of the parameters used for HPC controller.

In Table 5-8, the parameters C, b, τ and θ are computed based on the regression modeling of the HX process. The horizon time is chosen approximately from the settling time. Settling time is computed using Equation (5-2) and so 2/3rd of the settling time is approximately 65.39s. The horizon time at the minimum has to be at least 2/3rd of the settling time. For the HPC controller a value higher than 2/3rd of the settling time, 75s is chosen as the horizon interval.

$$\text{Settling Time} \approx 4\tau \tag{5-2}$$

The tuning parameter (period as in Equation (3-19)) is chosen approximately. The tuning parameter approximately has to be at least 0.8 times the τ. In this case a tuning parameter of 35 s is used which is higher than the approximate estimation. The best horizon interval and the tuning parameter for the controller is not estimated because the aim of this work is to demonstrate the usage of Leapfrogging optimization technique for a HPC application. Hence approximate values are used in for both the horizon interval and tuning parameter.

The parameter values are rounded to the second decimal place for convenience of further implementation of the parameter values for HPC control implementation.

Table 5-8: Controller Parameter Values (Static Delay)

Parameter	Value
C	0.03
b	1.37
τ	24.52 s
θ	9 s
Horizon	75 s
Tuning Parameter	35 s

5.3.2 HPC Controller With Dynamic Delay:

Table 5-9 lists the values of the parameters used for HPC controller.

In Table 5-9, the parameters C, b, τ and d are computed based on the regression modeling of the HX process. The horizon time is chosen approximately from the settling time. Settling time is computed using Equation (5-2) and so 2/3rd of the settling time is approximately 61.03 2. The horizon time at the minimum has to be at least 2/3rd of the setline time. For the HPC controller a value higher than 2/3rd of the settling time, 75s is chosen as the horizon interval.

Table 5-9: Controller Parameter Values (Dynamic Delay)

Parameter	Value
C	0.05
b	1.25
τ	22.88 s
d	0.58
Horizon	75 s
Tuning Parameter	35 s

The tuning parameter (period of Equation (3-19)) is chosen approximately. The tuning parameter approximately has to be at least 0.8 times the τ. In this case a tuning parameter of 35 s is used which is higher than the approximate estimation. Similar to the case of static delay, the best horizon interval and the tuning parameter for the controller is not estimated, approximate sensible values are used in for both the horizon interval and tuning parameter.

5.3.3 HPC Controller Results With Both Static and Dynamic Delay:

As discussed in Chapter 4 (Experimentation), several tests are conducted on the controller to test the working of the controller. Following are the tests and corresponding results for the HPC controller with static delay model and dynamic delay.

1. Bumpless Transfer:

Figure 5-6 demonstrates the set point tracking of in the MANUAL mode and the bumpless transfer from MANUAL to AUTO mode. From Figure 5-6, it can be seen that the HPC controller in MANUAL mode is continuously tracking the process value to avoid any bumps when switched to AUTO mode.

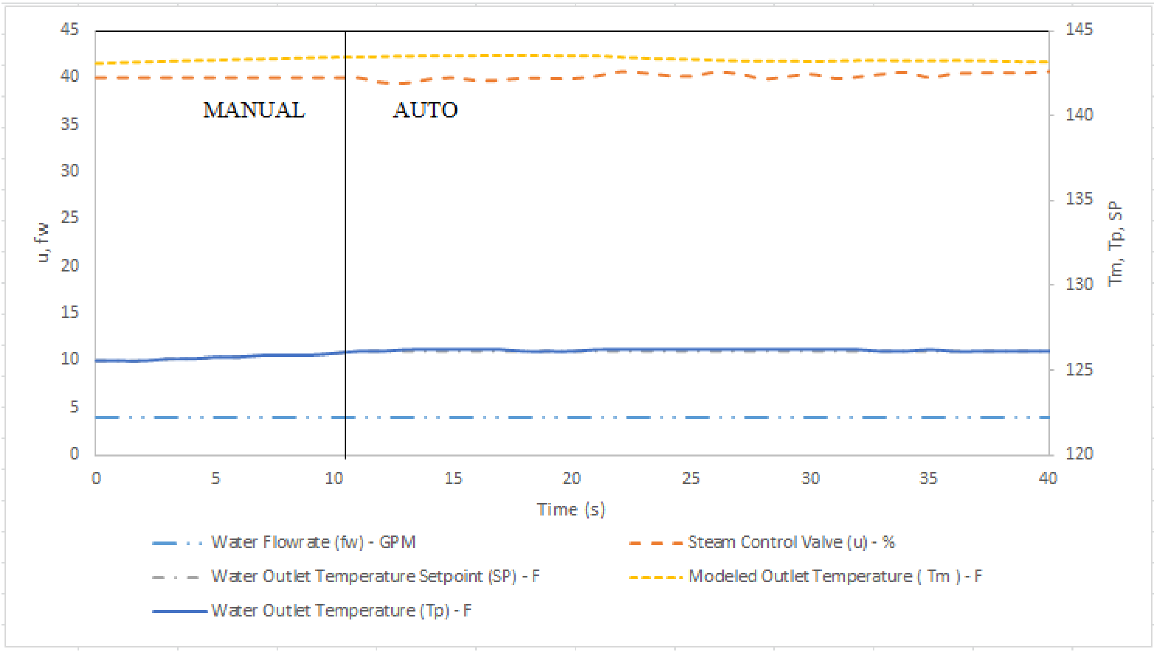


Figure 5-6: Bumpless Transfer (Static Delay)

At about 11s the controller is switched to AUTO mode. A bumpless transfer is seen at 11s due to set point tracking in the MANUAL mode. The HPC controller, in AUTO mode maintains the set point by making the model move along the reference trajectory which moves towards the biased set point.

Similarly, Figure 5-7 demonstrates the set point tracking of in the MANUAL mode and the bumpless transfer from MANUAL to AUTO mode in the HPC with dynamic delay. From Figure 5-6, it can be seen that the HPC controller in MANUAL mode is continuously tracking the process value to avoid any bumps when switched to AUTO mode.

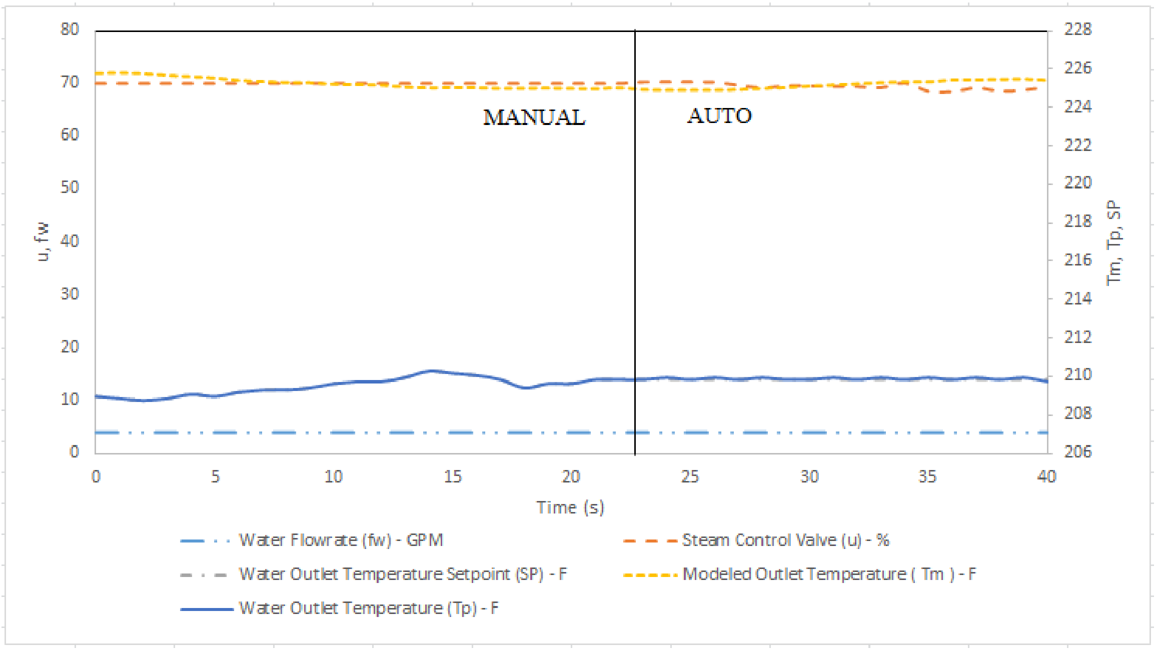


Figure 5-7: Bumpless Transfer (Dynamic Delay)

The controller is switched to AUTO mode at about 23s and a bumpless transfer of the process from the MANUAL to AUTO mode is seen.

Hence, in both the cases with static delay and the dynamic delay, the controller is able to transfer from MANUAL to AUTO mode without a bump.

2. Disturbance Rejection:

A set point change is made at about 25 s and the controller is allowed to reach steady state. Once steady state is attained the flowrate of water is varied from 4 GPM to 5 GPM, at about 209 s to test the disturbance rejection capability of the controller. Figure 5-8 shows the disturbance rejection capability of the controller. In Figure 5-8, the bottommost line of the graph (dashed line with two dots inbetween) represents the flow rate of water. As can be seen in Figure 5-8, once the process reaches the set point, a change in the flow rate of water at about 209 s did not affect the process being at the set point. The controller adjusted the MV to be able to sustain the process at the set point. The MV reacted to the disturbance after the delay which can be seen from the dotted line in Figure 5-8.

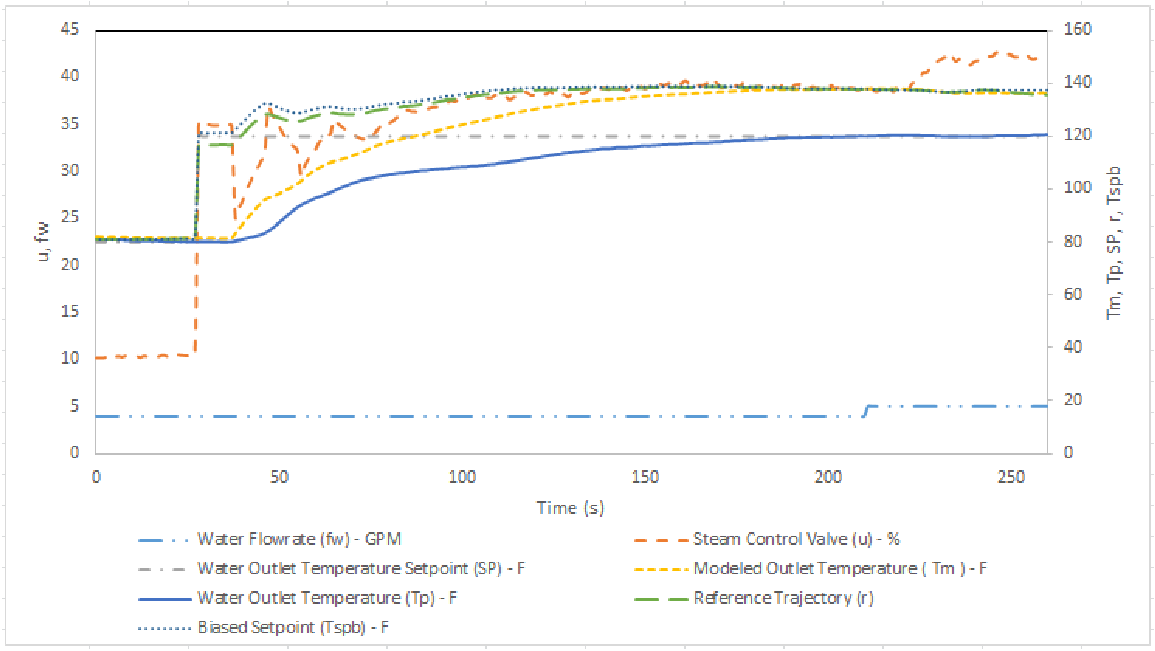


Figure 5-8: Disturbance Rejection (Static Delay)

Similarly, a set point change is made at about 9 s and the controller is allowed to reach steady state. Once steady state is attained the flowrate of water is varied from 4 GPM to 5 GPM, at about 224 s to test the disturbance rejection capability of the controller. Figure 5-9 shows the disturbance rejection capability of the controller with dynamic delay. Same as in Figure 5-8, the bottommost line of the graph (dashed line with two dots inbetween) represents the flow rate of water. The controller adjusted the MV to be able to sustain the process at the set point. The MV reacted to the disturbance after the delay which can be seen from the dotted line in Figure 5-9.

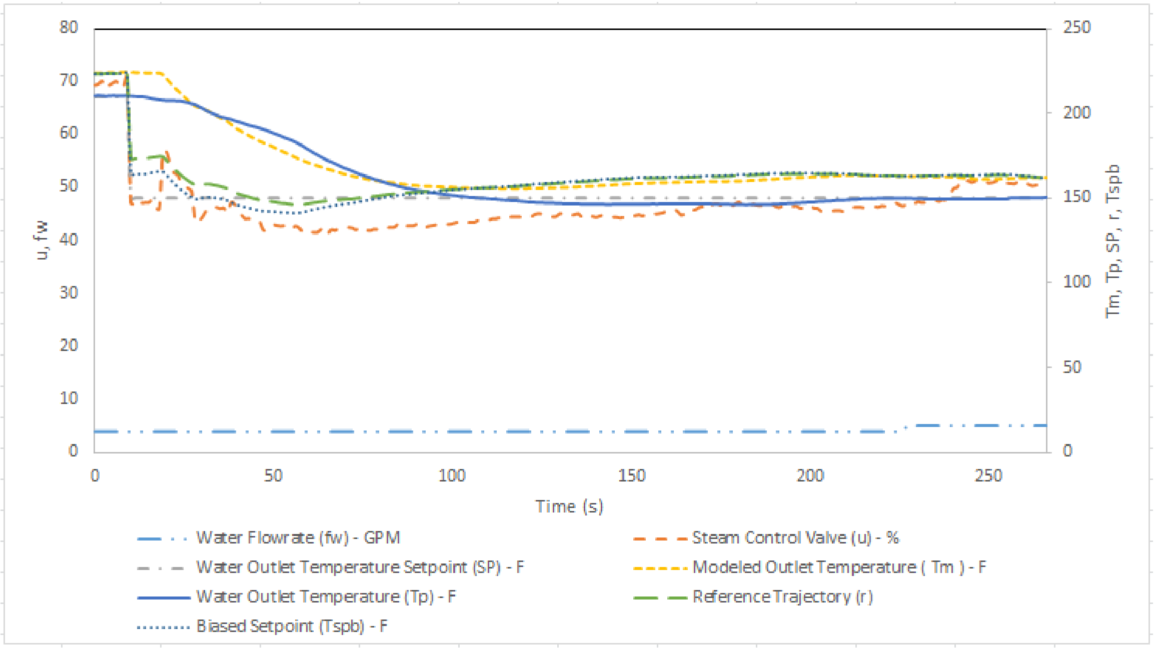


Figure 5-9: Disturbance Rejection (Dynamic Delay)

So, in both the cases with static delay and the dynamic delay, the controller is able to reject the disturbance caused.

3. Constraint Recovery:

The steam valve opening is constrained to have a value between 5 and 100 % to avoid any stiction caused when the valve goes below 5 %. In Figure 5-10, at about 10 s, a set point of 70 F is given to the controller. Due to the constraint, the controller makes the process reach a value of 76 F instead of 70 F. It can be seen from Figure 5-10 that when the set point is 70 F, the process could not reach the set point because of the constraint on MV which cannot go below 5%. So, at steady state the process is offset by about 4 F from the set point.

At about 128 s, the set point is moved to 100 F and it can be seen in Figure 5-10 that without wind up the controller immediately responds to the new set point and makes the process move towards and hold at 100 F.

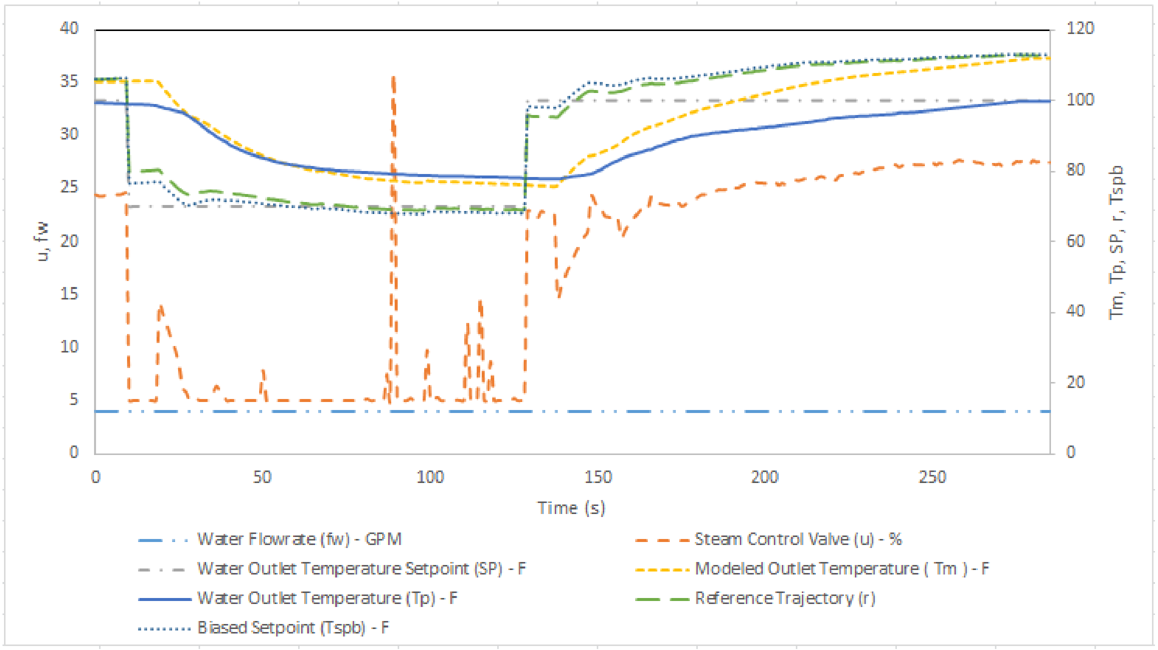


Figure 5-10: Constraint Recovery (Static Delay)

In Figure 5-11, at about 8 s, a set point of 70 F is given to the controller. The constraint on the MV makes the process reach a value of 76 F instead of 70 F. It can be seen from Figure 5-11 that when the set point is 70 F, the process could not reach the set point because of the constraint on MV which cannot go below 5%. So, at steady state the process is offset by about 4 F from the set point.

At about 143 s, the set point is moved to 100 F and it can be seen in Figure 5-11 that without wind up the controller immediately responds to the new set point and makes the process move towards and hold at 100 F.

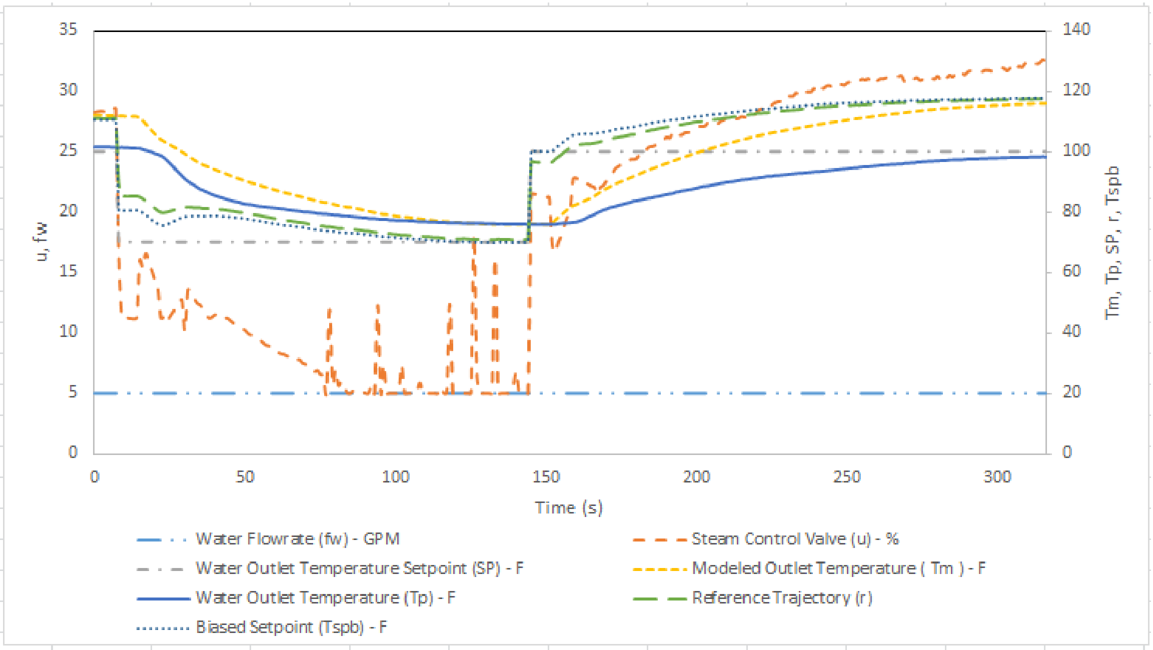


Figure 5-11: Constraint Recovery (Dynamic Delay)

Hence, in both the cases with static delay and the dynamic delay, the controller is able to reject recover from an unrealizable set point within the MV operational range.

4. Rate of Change Constraint Test:

The rate of change constraint on the controller is tested. To test this on the controller, initially the rate of change constraint is set to 100% which allows the consecutive MV moves to have a difference of either 100 or less than 100. In Figure 5-12, at about 8 s, the set point is changed to 160 F with a rate of change constraint of 100 %. Since the MV is allowed to make a change of either 100 % or less, the MV moves in Figure 5-12, is about 20 % .

At about 234 s, the set point is changed to 80 F with a rate of change constraint of 5 %. This means that consecutive MV values can differ either by 5 or less than 5. It can be seen in Figure 5-12, that the MV changes are 5 % or less as opposed to 20 % changes before. So, with rate of change constraint of 5 % the MV moves not more than 5 % at a time and makes the process move towards the set point.

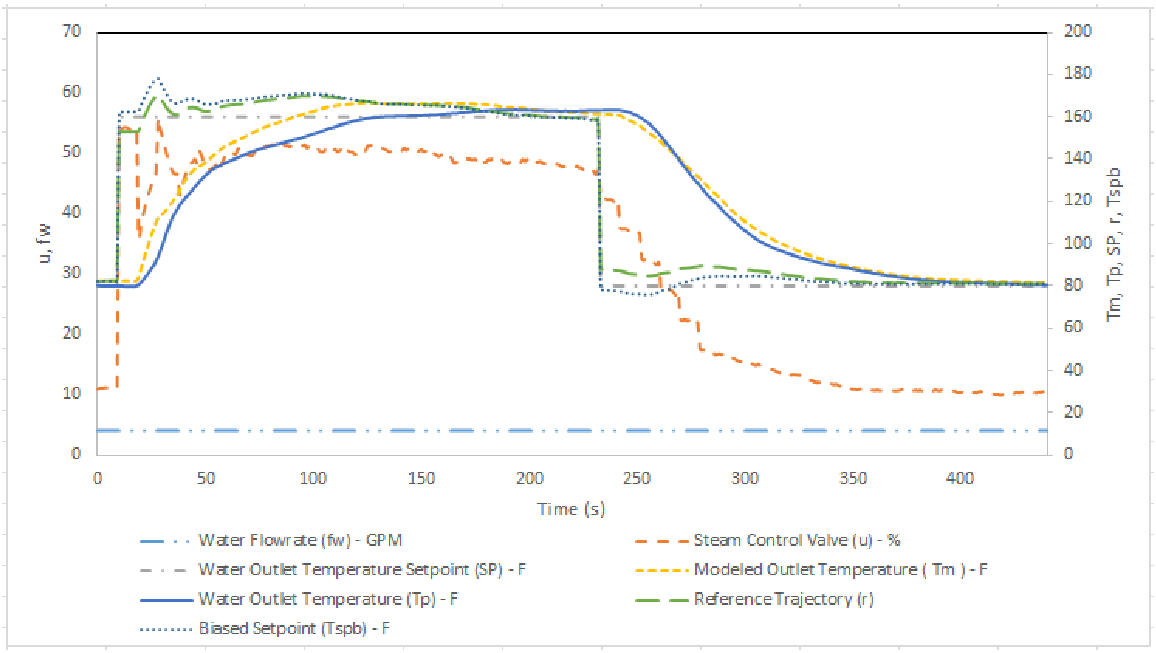


Figure 5-12: Rate of Change Constraint (Static Delay)

The rate of change constraint test is tested on the HPC with dynamic delay as well.

At about 10 s, the set point is changed to 150 F with a rate of change constraint of 100 %. This means that consecutive MV values can differ either by 100 % or less than 100 %. It can be seen in Figure 5-13, that the MV changes are about 10 %. At about 294 s, the set point is changed to 100 F with a rate of change constraint of 5 % which means consecutive MV moves can only have a difference of 5 or less. In Figure 5-13, it can be seen that from 294 s onwards because of the rate of change constraint of 5 %, the MV moves are 5 % or less. So, with rate of change constraint of 5 % the MV moves not more than 5 % at a time and makes the process move towards the set point.

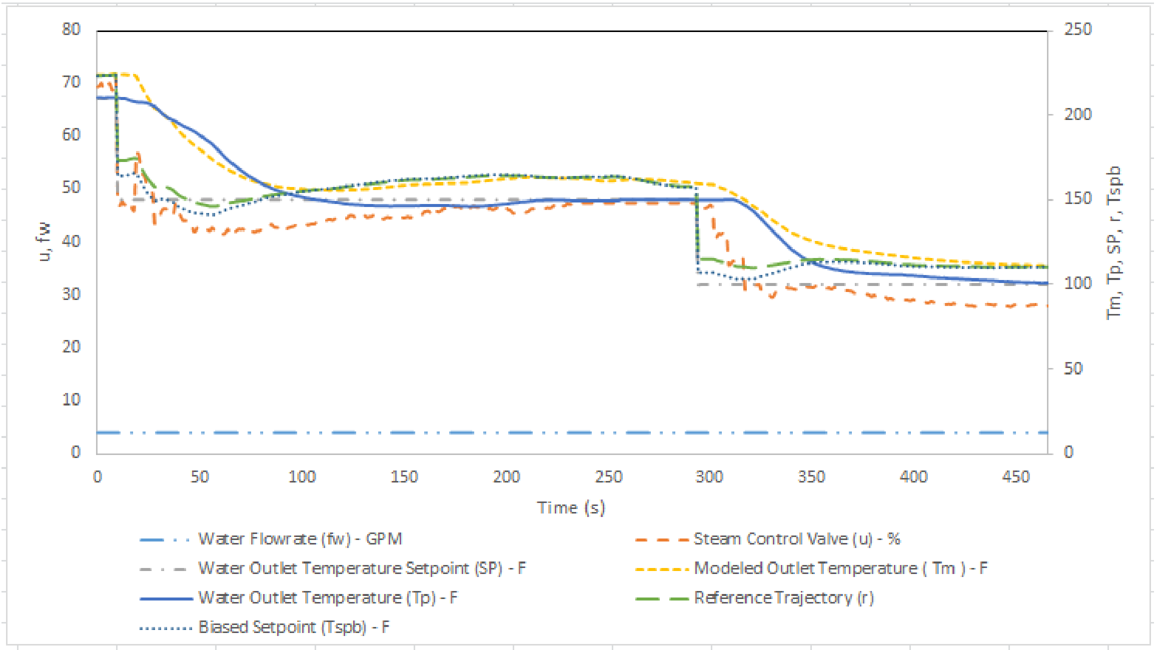


Figure 5-13: Rate of Change Constraint (Dynamic Delay)

So in both the cases, the controller takes into account the rate of change constraint while implementing control action.

5. Controller Tuning:

The controller is tested for different tuning parameters. Aggressive tuning parameter makes the controller aggressive making the reference trajectory move faster towards the biased set point while sluggish tuning parameter due to high tuning value makes the controller sluggish allowing the reference trajectory to take a little longer to reach the biased set point.

In Figure 5-14, at about 15 s, the set point is changed to 90 F. The tuning parameter used for this change is 10 s which is lesser than the tuning parameter used for the controller until now (35 s). The controller aggressiveness can be seen from the consecutive MV moves. The first MV move is about 40 % making it go to 5 %. After the 40 % move the controller realizes that if the process moves in the same manner, it would go past the set point and hence the MV back offs to 25 %. In response the process with a delay goes closer to the set point at about 50 s and then again moves farther from the set point. Because of low tuning parameter the controller tries to make the process reach the set point faster showing the aggressiveness of the controller.

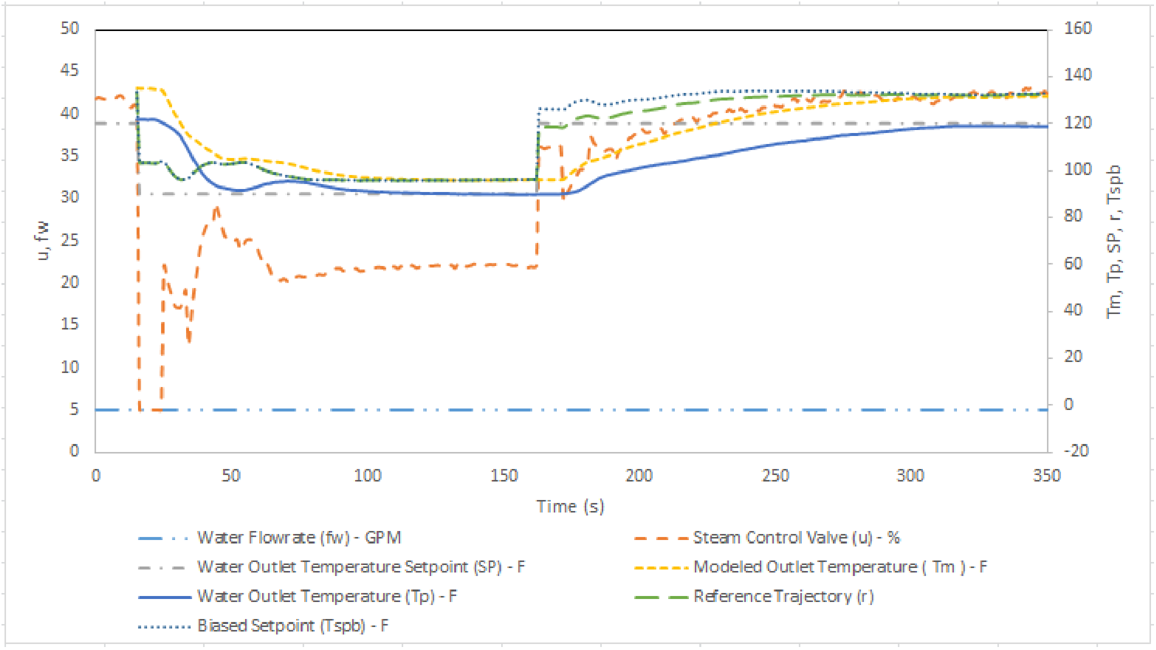


Figure 5-14: Controller Tuning (Static Delay)

At about 160 s, the set point is change to 120 F. The tuning parameter is changed to 55 s. As can be seen in Figure 5-14, the process after 160 s, moves slowly towards the set point. It took the process about 160 s with a tuning parameter of 55s to first get closer to the set point (about 119 F) whereas in the case of tuning parameter with 10 s, it took about 40 s to first get closer to the set point (about 91 F). Figure 5-14, thus shows the aggressive and sluggish behavior of the controller with different tuning parameters.

Similarly, different controller tuning parameters are included in the HPC controller with dynamic delay. In Figure 5-15, at about 10 s the set point is changed to 150 F with a tuning parameter of 10 s. The low tuning parameter makes the controller aggressive, by pushing the process towards the set point faster. As soon as the set point change is made MV changes about 50 % trying to push the process faster and so the process reaches the set point in about 20 s, but then over shoots beyond the set point. Controller accordingly backs off's making the process move towards set point. Again the controller pushes hard making the process value again go past the set point at about 92 s. The controller and the process keep oscillating because of the aggressiveness of the controller.

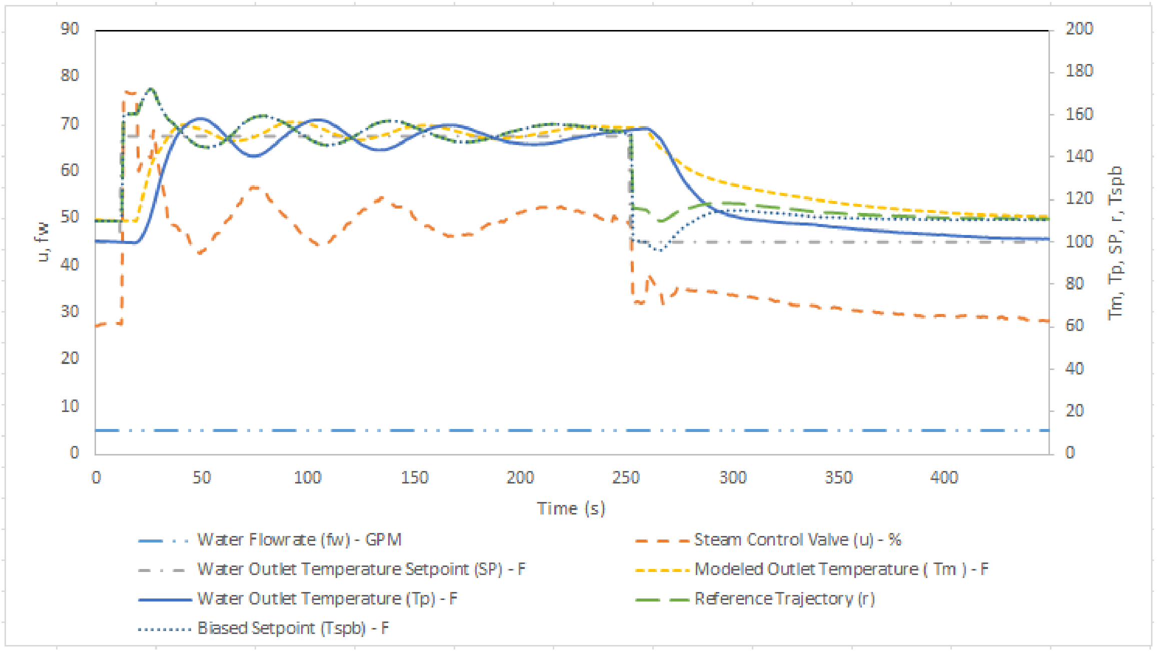


Figure 5-15: Controller Tuning (Dynamic Delay)

At about 253 s, the set point is changed to 100 F but with a tuning parameter of 60 s. As opposed to the situation with a tuning parameter of 10 s, the controller gradually moves the process towards the set point. The MV moves are gradual. The process first reaches the set point in about 200 s. After reaching the set point, the controller sustains the process to stay at the set point.

In both the cases, tuning parameter of 10 s (lower than 35s and 35s) makes the controller aggressive. Tuning parameters of 55s and 60 s (greater than 35 s and 35 s) makes the controller sluggish by moving the process slowly towards the set point.

5. Controller Behavior:

Figure 5-16 shows the behavior of the controller with static delay model. The controller aims at moving the process towards the set point and holding at the set point. A down step change to 90 F is made to the set point at about 5s. The reference trajectory moved towards the biased set point. The model following the reference trajectory path made the process move towards and hold at 90F.

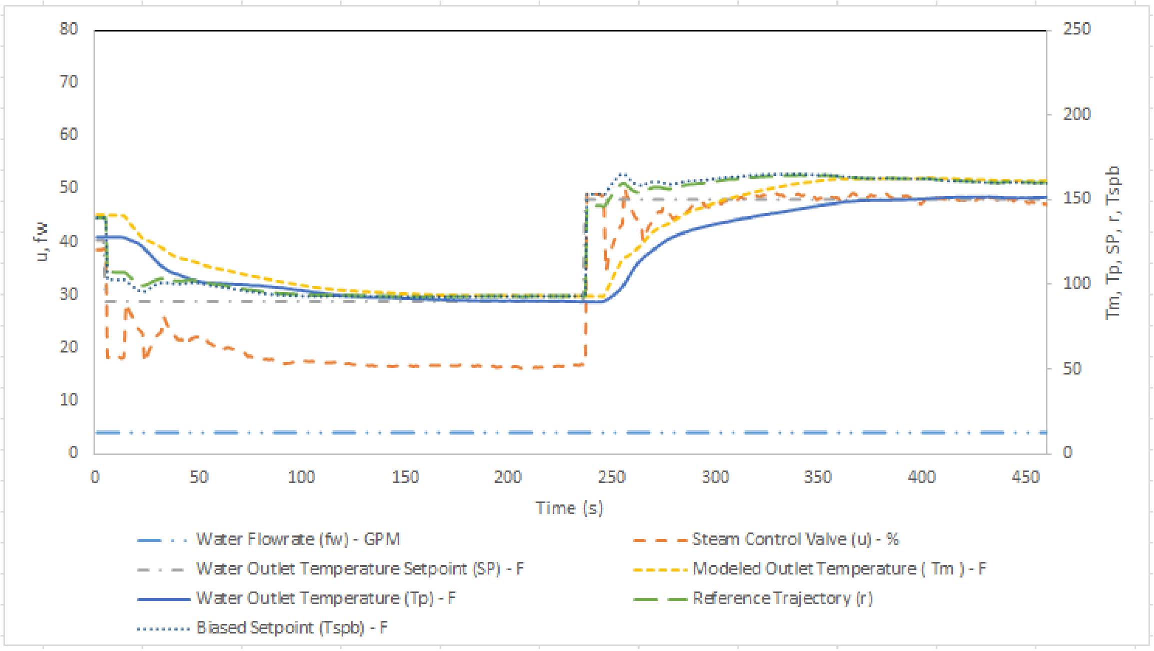


Figure 5-16: Controller Behavior (Static Delay)

In Figure 5-16 another step up to the set point is made at about 237 s where the set point is moved to 150 F. Again the controller makes the process move towards and hold at 150 s. The process started reacting to the set point change at 237 s at about 246 s showing the delay is about 9 s.

Figure 5-17 shows the behavior of the controller with dynamic delay. Similar to the controller behavior with static delay, the controller with dynamic delay also aims at moving the process towards the set point and holding at the set point. The set point in this case to be able to compare with the static delay case is changed to 90 F at about 9 s and once the process reached steady state the set point is changed to 150 F. When moving down to 90 F the process did not over shoot in the case of dynamic delay, but when moving towards 150 process did over shoot.

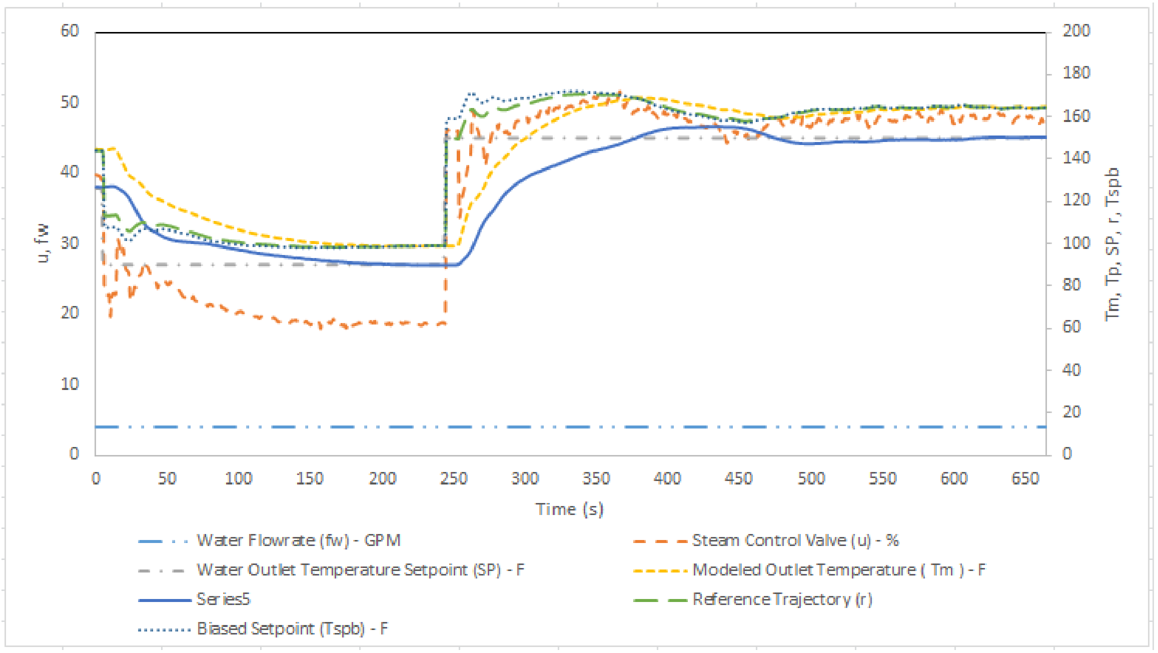


Figure 5-17: Controller Behavior (Dynamic Delay)

The goodness of the controller is tested in both the cases of static delay and dynamic delay. The goodness of the controller is tested using two scales. One scale measures the integral square error (ISE) as shown in Equation (5-3) between the process value and the set point and the other scale measures the total travel of the MV as shown in Equation (5-4) to make the process reach the set point.

$$ISE = \sum (SetPoint - ProcessValue)^2 \quad (5\text{-}3)$$

$$MVTravel = \sum Abs(MV_{new} - MV_{old}) \quad (5\text{-}4)$$

The summation in both Equation (5-3) and Equation (5-4) is computed across the time the set point is changed to the time the process reaches steady state. Table 5-10 lists the comparison between the HPC controllers with static delay and dynamic delay model. In both the cases of set point the controller with dynamic delay model has a lower ISE than the controller with the static model. But, the MV travel for the dynamic model is higher compared to the static model controller.

Table 5-10: Comparison of Controller Behavior With Static and Dynamic Model

	Static Delay Model	Dynamic Delay Model
To set point 90 F (ISE)	10228.31	10188.04
To set point 90 F (MVTravel)	80.13	88.28
To set point 150 F (ISE)	103558.6	29357.13
To set point 150 F (MVTravel)	123.40	184.24

CHAPTER 6

CONCLUSION AND FUTURE WORK

6.1 Conclusion:

Following are the conclusions from this study:

1. Based on the fundamental knowledge that presence of at least one player in the global attractor area would pull all the other players towards it, helped suggest the idea of improved initialization which leads to OF surface exploration before executing the Leapfrogging optimization technique.
2. Leapfrogging with improved initialization demonstrated increase in the probability of finding the global optimum in many test functions with multiple optima. The improvement also demonstrated increased efficiency (low PNOFE) of the optimization technique. In the case of a single optimum function, Leapfrogging with improved initiation is not effective as there is no need for surface exploration.
3. For the Leapfrogging optimization with improved initialization *a priori* knowledge about the global attractor Region A as a fraction of the entire DV space is needed. Perfect *a priori* knowledge is not required. An approximate yet sensible knowledge is sufficient.
4. Leapfrogging and Leapfrogging with improved initialization are demonstrated in regression steady state modeling of a HX process by generating a skyline function of the

process. Two steady state models, one with static delay and the other with dynamic delay are developed. The models generated are simple to be able to use for implementing HPC on the HX. This is a case with no prior *a priori* knowledge about the OF surface. Leapfrogging with improved initialization proved its ability to find the global optimum with a higher probability in both the cases. However, the PNOFE is higher because of the initial surface exploration in both the cases. The dynamic model of the HX process is built from the steady state model.

5. Usage of Leapfrogging for implementing nonlinear HPC on the pilot scale HX equipment is demonstrated, paving way for the possibility of using Leapfrogging optimization technique in chemical industries.

6.2 Future Work [1]:

Following are the possibilities for future exploration in Leapfrogging optimization:

1. Computing the right number of players per dimension. Currently 10 players per dimension are used which is estimated through experiments. A mathematical support for the right number of players per dimension can be explored.
2. Exploring the computation of global attractor Region A for higher dimensions. For computing the total Region A in all dimensions, this study uses Region A in a dimension raised to the power of number of dimensions. Further studies exploring a better computation of Region A in all dimensions can further reduce the PNOFE.
3. Defining an iteration. Depending upon the definition of an iteration the PNOFE of the optimization technique varies. Should an iteration be considered as a leap over? Should an iteration be considered as a leap over in all the dimensions? Should it include the leap overs that place the player in constrained region? Consider defining an iteration for comparison purposes with other optimization techniques.

4. Exploring the possibility of starting with particle swarm optimization technique and switching to Leapfrogging as an end-game strategy. Particle swarm optimization technique can be used for initial surface exploration and once the global optimum is surrounded then switch to Leapfrogging. This is expected to improve the probability of reaching the global with less number of function evaluations.
5. Changing the axes of the leap-to window heuristically based on the location of the best and the worst player. Currently the leap-to window is always rectangular. Consider different axes be used for the leap-to window.
6. Reducing the number of active players during end-game. Consider gradually reducing the number of active players with approaching the global to further reduce the PNOFE.
7. Adding momentum while leaping over to carry forward the movement trend of the best player. Consider adding a momentum factor during every leap over so that the trend of the best player is carried.
8. Preserving the worst player locations. Currently the trend of the worst player locations is not preserved. Should the locations be preserved? Will preserving help quicker convergence? Will preserving help come up with the better repulsion technique to avoid any other player to leap into the region where a worst player was present before? Consider preserving the worst player locations to avoid players fortuitously fall into the same region again during leap overs.

REFERENCES

1. Rhinehart, R.R., Su, M., and Manimegalai-Sridhar, U., *Leapfrogging and synoptic Leapfrogging: A new optimization approach.* Computers & Chemical Engineering, 2012. 40(0): p. 67-81.
2. Raul, P.R., Srinivasan, H., Kulkarni, S., Shokrian, M., Shrivastava, G., and Rhinehart, R.R., *Comparison of Model-Based and Conventional Controllers on a Pilot-Scale Heat Exchanger*. ISA Transactions, 2013. 52: p. 391-405.
3. Manimegalai Sridhar, U., *Pseudo-component viscoelastic modeling of soft tissues*, 2010, Oklahoma State University: United States -- Oklahoma. p. 99.
4. Ratakonda, S., Manimegalai-Sridhar, U., Rhinehart, R.R., and Madihally, S.V., *Assessing viscoelastic properties of chitosan scaffolds and validation with cyclical tests.* Acta Biomaterialia, 2012. 8(4): p. 1566-1575.
5. Ratakonda, S., *Assessing viscoelastic properties of chitosan scaffolds and validating sequential and cyclical tests*, 2011, Oklahoma State University: Ann Arbor. p. 105.
6. Sethuraman, V., *Viscoelastic modeling of stress relaxation behavior in biodegradable polymers*, 2013, Oklahoma State University: Ann Arbor. p. 83.
7. Chen, H., *Incorporation of the generalized TSK models in model predictive control*, 2011, Oklahoma State University: United States -- Oklahoma. p. 85.
8. Manimegalai-Sridhar, U., Govindarajan, A., and Rhinehart, R.R., *Improved initialization of players in leapfrogging optimization.* Computers & Chemical Engineering, 2014. 60: p. 426-429.

9. Beigler, L.T., and Grossman, I. E., *Retrospective on optimization.* Computers & Chemical Engineering, 2004. 28: p. 1169-1192.

10. Babu, B.V. and Angira, R., *Modified differential evolution (MDE) for optimization of non-linear chemical processes.* Computers & Chemical Engineering, 2006. 30(6–7): p. 989-1002.

11. Golshan, M., Pishvaie, M. R., and Boozarjomehry, R. B., *Stochastic and global real time optimization of Tennessee Eastman challenge problem.* Engineering Applications of Artificial Intelligence, 2008. 21: p. 215 - 228.

12. Hsu, C.-C., and Lin, G.-Y., *Digital redesign of uncertain interval systems based on time response resemblance via particle swarm optimization.* ISA Transactions, 2009. 48: p. 264-272.

13. Lee, J.H.V.d., Svrcek, W. Y., and Young, B. R., *A tuning algorithm for model predictive controllers based on genetic algorithms and fuzzy decision making.* ISA Transactions, 2008. 47: p. 53-59.

14. Li, J., and Rhinehart, R. R., *Heuristic random optimization.* Computers and Chemical Engineering Applications of Artificial Intelligence, 1998. 22(427-444).

15. Ochoa, S., Wozny, G., and Repkec, J.-U., *A new algorithm for global optimization: Molecular-Inspired Parallel Tempering.* Computers and Chemical Engineering, 2010. 34: p. 2072-2084.

16. Ramezani, M.H., and Sadati, N., *Hierarchical optimal control of large-scale nonlinear chemical processes.* ISA Transactions, 2009. 48: p. 38-47.

17. Singh, V., Gupta, I., and Gupta, H. O., *ANN based estimator for distillation–inferential control.* Chemical and Engineering Processing, 2005. 44: p. 785-795.

18. Storn, R. and Price, K., *Differential Evolution - A Simple and Efficient Heuristic for global Optimization over Continuous Spaces.* Journal of Global Optimization, 1997. 11(4): p. 341-359.

19. Cordeau, J.-F., Gendreau, M., Laporte, G., Potvin, J.-Y., and Semet, F., *A guide to vehicle routing heuristics.* Journal of the Operational Research Society, 2002. 53: p. 512-522.

20. Derigs, U., and Kaiser, R., *Applying the attribute based hill climber heuristic to the vehicle routing problem.* European Journal of Operational Research, 2007. 177: p. 719-732.

21. Rhinehart, R.R., Su, M., and Manimegalai-Sridhar, U., *Leapfrogging and synoptic Leapfrogging: A new optimization approach.* Computers and Chemical Engineering, 2012. 40: p. 67-81.

22. Richalet, J., Rault, A., Testud, J.L., and Papon, J., *Model predictive heuristic control: Applications to industrial processes.* Automatica, 1978. 14(5): p. 413-428.

23. Young, R.E., Bartusiak, R.D., and Fontaine, R.W., *Evolution of an industrial nonlinear model predictive controller*. in *AICHE SYMPOSIUM SERIES*. 2002. New York; American Institute of Chemical Engineers; 1998.

24. Morari, M. and Lee, J. H. , *Model predictive control: past, present and future.* Computers & Chemical Engineering, 1999. 23(4–5): p. 667-682.

25. Bauer, M. and Craig, I.K. , *Economic assessment of advanced process control–A survey and framework.* Journal of Process Control, 2008. 18(1): p. 2-18.

26. Alessio, A. and Bemporad, A., *A Survey on Explicit Model Predictive Control*, in *Nonlinear Model Predictive Control*, L. Magni, D. Raimondo, and F. Allgöwer, Editors. 2009, Springer Berlin Heidelberg. p. 345-369.

27. Lee, J., *Model predictive control: Review of the three decades of development.* International Journal of Control, Automation and Systems, 2011. 9(3): p. 415-424.

28. Pardalos, P., Prokopyev, O., and Busygin, S., *Continuous Approaches for Solving Discrete Optimization Problems*, in *Handbook on Modelling for Discrete Optimization*, G. Appa, L. Pitsoulis, and H.P. Williams, Editors. 2006, Springer US. p. 39-60.

29. Lenstra, J.K., Kan., A. H. G., and Schrijver, A., *History of Mathematical Programming: A Collection of Personal Reminiscences.* 1991.

30. Edgar, T.F., Himmelblau, D.M., and Ladson, L.S., *Optimization of Chemical Process*. 2001, New york: McGraw-Hill. 651.

31. Binder, T., Blank, L., Bock, H. G., Bulitsch, R., Dahmen, W., Diehl, M., Kronseder, T., Marquardt, W., Schloeder J., and von Stryk, O., *Introduction to model based optimization of chemical processes on moving horizons.* Online optimization of large scale systems, 2001.

32. Schittkowski, K., *More test examples for nonlinear programming codes.* Lecture notes in economics and mathematical systems, 1987. 282.

33. Lucia, A. and Xu, J., *Methods of successive quadratic programming.* Computers & Chemical Engineering, 1994. 18, Supplement 1(0): p. S211-S215.

34. Han, S.-P., *Superlinearly convergent variable metric algorithms for general nonlinear programming problems.* Mathematical Programming, 1976. 11(1): p. 263-282.

35. Powell, M.J.D., *A fast algorithm for nonlinearly constrained optimization calculations*, in *Numerical Analysis*, G.A. Watson, Editor. 1978, Springer Berlin Heidelberg. p. 144-157.

36. Moré, J., *The Levenberg-Marquardt algorithm: Implementation and theory*, in *Numerical Analysis*, G.A. Watson, Editor. 1978, Springer Berlin Heidelberg. p. 105-116.

37. Kolda, T.G., Lewis., R.M., and Torczon, V., *Optimization by Direct Search: New Perspectives on Some Classical and Modern Methods.* Society for Industrial and Applied Mathematics, 2003. 45.

38. Duan, Q., Liao., W.L. and Yi, H.Z., *A comparative study of different local search application strategies in hybrid metaheuristics.* Applied Soft Computing, 2013. 13: p. 1464-1477.

39. Yi, H., Duan, Q., Warren, and Liao, T.W., *Three improved hybrid metaheuristic algorithms for engineering design optimization.* Applied Soft Computing, 2013. 13: p. 2433-2444.

40. Kennedy, J., and Eberhart, R. C., *Particle swarm optimization.* Proceedings of the 1995 IEEE International Conference on Neural Networks, 1995: p. 1942-1948.

41. Yuzgec, U., *Performance comparison of differential evolution techniques on optimization of feeding profile for an industrial scale baker's yeast fermentation process.* ISA Transactions, 2010. 49: p. 167-176.

42. Burke, E.K., Gustafson, S., and Kendall, G., *Diversity in genetic programming: an analysis of measures and correlation with fitness.* IEEE Transatctions on Evolutionary Computation, 2004. 8: p. 47-62.

43. Chen, G., Low, C. P., and Yang, Z., *Preserving and exploiting genetic diversity in evolutionary programming algorithms.* IEEE Transactions on Evolutionary Computation, 2009. 13: p. 661-673.

44. Leung, Y., Gao, Y., and Xu, Z.-B., *Degree of population diversity – a perspective on premature convergence in genetic algorithms and its Markov chain analysis.* IEEE Transactions on Neural Networks, 1997. 8: p. 1165-1176.

45. Nelder, J.A., and Mead, R., *A simplex method for function minimization.* The Computer Journal, 1965. 7: p. 308-313.

46. Spendly, W., Hext, G. R., and Himsworth, F. R., *Sequential application of simplex designs in optimization and evolutionary operation.* Techmometrics, 1962. 4: p. 441-461.

47. Iyer, M.S. and Rhinehart, R.R., *A method to determine the required number of neural-network training repetitions.* Neural Networks, IEEE Transactions on, 1999. 10(2): p. 427-432.

48. Cutler, C.R. and Ramaker, B., *Dynamic matrix control—a computer control algorithm*. in *Joint Automatic Control Conference*. 1980.

49. Camacho, E.F., and Bordons Alba, C., *Model Predictive Control in the Process Industry*, ed. Springer-Verlag. 1995, Berlin, Germany.

50. Gu, R., Bhattacharyya, S.S., and Levine, W.S., *Methods for Efficient Implementation of Model Predictive Control on Multiprocessor Systems*, in *2010 IEEE International Conference on Control Applications*2010: Yokohama, Japan.

51. Baskar, L.D., Schutter, B.D. and Hellendoorn, H., *Model predictive control for intelligent speed adaptation in intelligent vehicle highway systems*. Proceedings of the 17th IEEE International Conference on Control Applications, 2008: p. 468-473.

52. Maciejowski, J.M., *Predictive Control with Constraints*, ed. P. Hall. 2002, Harlow, England.

53. Rhinehart, R.R., *Advanced Process Control Lecture Notes*. Spring 2013, Oklahoma State University.

54. Govindarajan, A., Jayaraman, S.K., Sethuraman, V., Raul, P., and Rhinehart, R., *Cascaded process model based control:Packed absorption column application.* ISA Transactions, 2013.

55. Smith, C.A., Corripio, A., *Principles and practise of automatic process control.* 1997.

56. Camacho, E.F., *Model Predcitive Control*. Second ed. Advanced Textbooks in Control and Signal Processing. 2004: Springer-Verlag. 405.

57. Wang, Y. and Boyd, S., *Fast model predictive control using online optimization*. in *Proceedings of the IFAC World Congress*. 2008. Seoul.

58. Rhinehart, R.R., Gebreyohannes, S., Manimegalai-Sridhar, U., Patrachari, A., and Rahaman, M.S., *A power law approach to orifice flow rate calibration.* ISA Transactions, 2011. 50: p. 329-341.

APPENDICES

Appendix A: Improved initialization leapfrogging

The VBA Code for Leapfrogging optimization technique is present in Appendix A of the original publication on Leapfrogging [1]. The improved initialization is a modified version of the already available Leapfrogging code. The modification is done in the InitializeOptimizer sub program. The modified version is as shown below:

```
Sub InitializeOptimizer ()
Initializations = [input appropriate value]          ‘Initial players required for surface exploration
NumTeammat es = [input approp riat e value]     ‘Final set of players for Leapfrogging
For PlayerNumber = 1 To NumTeammates
constraint = "Unassessed"
Do Until constraint = "PASS " ' each must be in a feasible location, repeat if not
For DVNum ber = 1 To DVDimension
PlayerPosition(DVNumber, PlayerNumber) = [appropriat e expression for the player ]
Next DVNumber
Call ConstraintTest
Loop
Call OFCalculate
PlayerOFValue(PlayerNumber) = OF
Next PlayerNumber
Call Find_High
```

```
PlayerNumber = NumTeammates + 1

For ExtraPlayer = 1 To Initializations – NumTeammates

constraint = "Unassessed"

Do Until constraint = "PASS"          'each must be in a feasible location, repeat if not

For DVNum ber = 1 To DVDimension

PlayerPosition(DVNumber, PlayerNumber) = [appropriat e expression for the player ]

Next DVNumber

Call ConstraintTest

Loop

Call OFCalculate

PlayerOFValue(PlayerNumber) = OF

If PlayerOFValue(PlayerNumber) < OFhigh Then  'new one better than existing worst – replace

For DVNum ber = 1 To DVDimension

PlayerPosition(DVNumber, PlayerNumber) = [appropriat e expression for the player ]

Next DVNumber

PlayerOFValue(LFHighpn) = PlayerOFValue(PlayerNumber)

Call Find_High

End If

Next ExtraPlayer

Call Find_Low

End Sub
```

Appendix B: HX Modeling:

The VBA codes for the HX modeling is built from the Leapfrogging base code as shown in the Appendix of original Leapfrogging publication [1]. The Find_High and Find_Low sub programs are the same. The Leapfrogging sub program is modified to call Assign before ContraintTest sub program and call T_Model before computing the OFCalculate sub program. The InitializeOptimizer sub program in Appendix A is modified to call the Assign and T_Model sub programs before computing the OF value. The VBA sub programs rest of the sub programs are shown below:

```
Sub main()
For TrialNumber = StartNum To EndNum
Call InitializeOptimizer
PlayerNumber = LFLowpn
Call T_Model
Call DataOut
For Iteration = 1 To MaxIter
For SubIteration = 1 To DVDimension 'Each iteration permits one leapover per DVDimension
Call Leapfrogging
Call RandomSubsetSS
If RStatistic < RCritical And Iteration > 50 Then Exit For
Next Iteration
Next TrialNumber
End Sub
*************************************************
Sub Assign()
Bmodel = PlayerPosition(1, PlayerNumber)
Cmodel = PlayerPosition(2, PlayerNumber)
```

```
Dmodel = PlayerPosition(3, PlayerNumber)

Tmodel = PlayerPosition(4, PlayerNumber)

End Sub

*************************************************

Sub T_Model()

ymodel(50) = ydata(50)

For DataNumber = 51 To Ndata

influence2 = udata(2, DataNumber)              'flow rate

Ndelay = Int(Dmodel / dt + 0.5)                 ‘static delay

Ndelay = Int(Dmodel / influence2/ dt + 0.5)     ‘dynamic delay

If Ndelay < 0 Then Ndelay = 0

If Ndelay > 50 Then Ndelay = 50

influence1 = udata(1, DataNumber - Ndelay)

influence2 = udata(2, DataNumber - Ndelay)

influence3 = udata(3, DataNumber - Ndelay)

ymodel(DataNumber) = (1 - dt / Tmodel) * ymodel(DataNumber - 1) + (dt / Tmodel) *
                     (influence3 + Cmodel * (influence1 ^ Bmodel) / influence2)

Next DataNumber

End Sub

*************************************************

Sub ConstraintTest()

constraint = "PASS"

If Cmodel < 0 Then constraint = "FAIL"

If Bmodel < 0 Then constraint = "FAIL"

If Dmodel < 0 Then constraint = "FAIL"

If Tmodel < 10 Then constraint = "FAIL"

End Sub
```

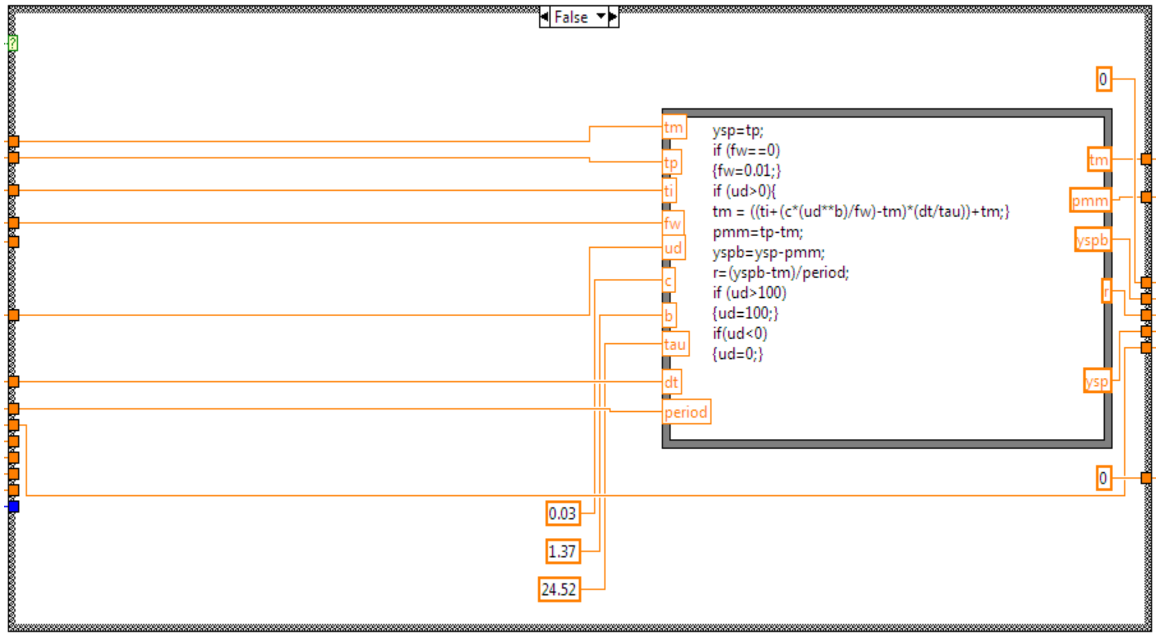
```
************************************************
Sub OFCalculate()
SSD = 0
For DataNumber = 100 To Ndata
SSD = SSD + (ydata(DataNumber) - ymodel(DataNumber)) ^ 2
Next DataNumber
RMS = Sqr(SSD / (Ndata - 100))
End Sub
************************************************
Sub RandomSubsetSS()
If Iteration = 1 Then
RRMSfilter = 0
varnum = 0
vardeno = 0
RRMSold = 0
NRSsets = Int(Ndata / 3 + 0.5)
End If
PlayerNumber = LFLowpn
Call Assign
Call T_Model
For RSSIndex = 1 To Ndata
RSdata(RSSIndex) = "open"
Next RSSIndex
SSDRSS = 0
For SelectedSetNumber = 1 To NRSsets
status = "taken"
```

```
Do Until status = "open"

RSSIndex = Ndelay + Int((Ndata - Ndelay) * Rnd() + 0.5)

status = RSdata(RSSIndex)

Loop

RSdata(RSSIndex) = "taken"

SSDRSS = SSDRSS + (ydata(RSSIndex) - ymodel(RSSIndex)) ^ 2

Next SelectedSetNumber

RRMS = Sqr(SSDRSS / NRSsets)

If Iteration = 1 Then RRMSfilter = RRMS

varnum = 0.05 * (RRMS - RRMSfilter) ^ 2 + 0.95 * varnum

RRMSfilter = 0.05 * RRMS + 0.95 * RRMSfilter

vardeno = 0.05 * (RRMS - RRMSold) ^ 2 + 0.95 * vardeno

RRMSold = RRMS

If vardeno <> 0 Then RStatistic = (2 - 0.05) * varnum / vardeno

End Sub
```

In Appendix B, the T_Model sub program includes both the static delay and dynamic delay computation lines. To use the dynamic delay, the static delay line is commented and vice versa.

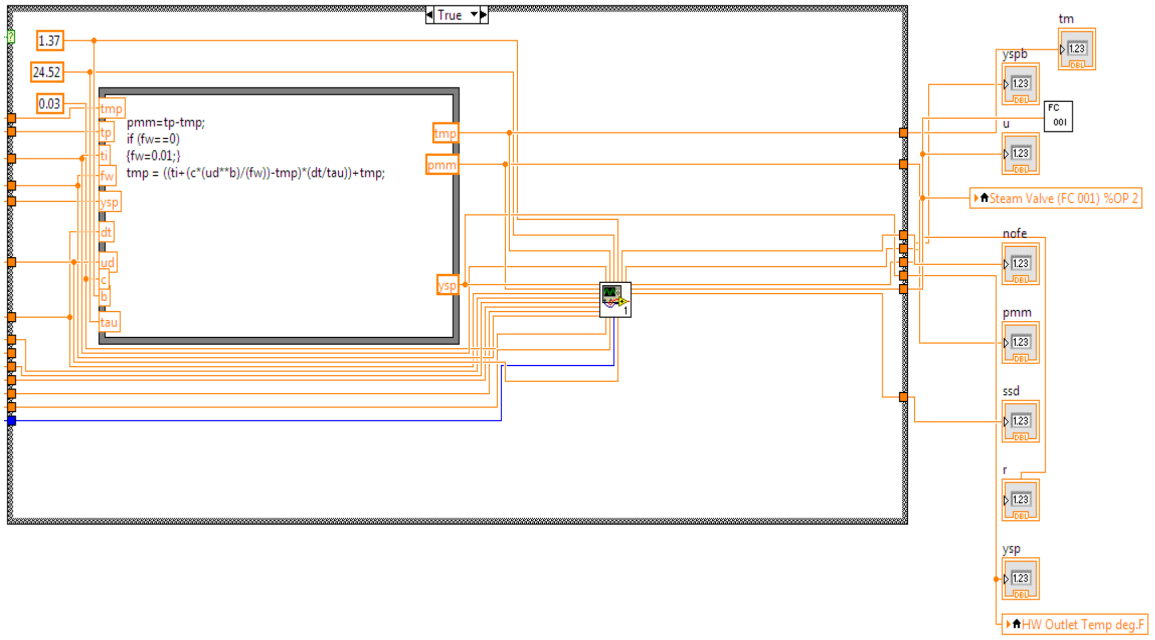
Appendix C: HPC Implementation:

The HPC is programmed in LabView. The HPC is built on the already existing LabView program for HX 001. The screenshot of the HPC MANUAL mode is shown in Figure C-1.



```
ysp=tp;
if (fw==0)
{fw=0.01;}
if (ud>0){
tm = ((ti+(c*(ud**b)/fw)-tm)*(dt/tau))+tm;}
pmm=tp-tm;
yspb=ysp-pmm;
r=(yspb-tm)/period;
if (ud>100)
{ud=100;}
if(ud<0)
{ud=0;}
```

Figure C-1: Screenshot of MANUAL Mode



```
pmm=tp-tmp;
if (fw==0)
{fw=0.01;}
tmp = ((ti+(c*(ud**b)/(fw))-tmp)*(dt/tau))+tmp;
```

Figure C-2: Screenshot of the AUTO Mode

The screenshot of the AUTO mode is as shown in Figure C-2. The inputs for both the AUTO mode and the MANUAL mode are obtained correspondingly from the already existing LabView program. The inputs specific to the HPC are given outside the case structure shown above. In Figure C-2 the formula node within the case structure is where the P2N model is computed. The VI present within the case structure is where the N2F and optimization computation takes place called the LeapfrogOpti VI. Figure C-3 shows the screen shot of the LeapfrogOpti VI.

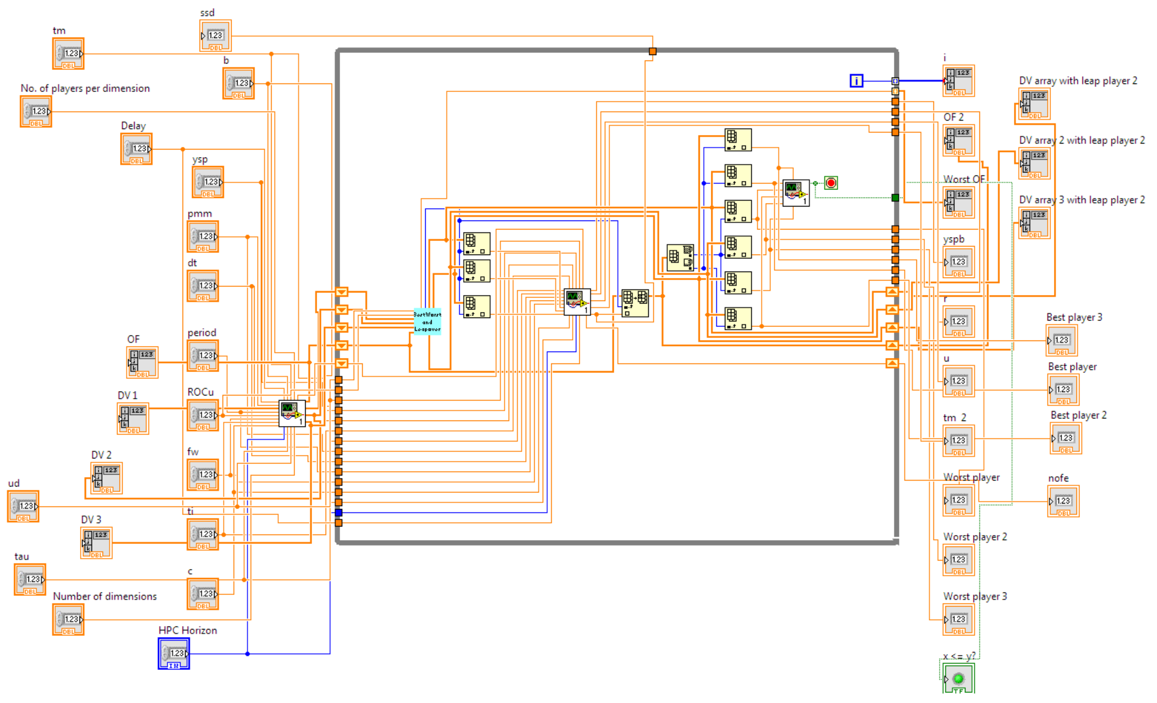


Figure C-3: Screenshot of the LeapfrogOpti VI

The VI present in the farther left is where the initialization of the players across the horizon is done. Figure C-4 is the screenshot of the player initialization VI. The rate of change constraint and the constraint on the MV are tested during the initialization stage. The VI present within the inner loop is for checking the rate of change and MV constraint. The VI present within the outer loop if for computing the N2F model for each set of initialization. Figure C-5 is the screenshot of the constraint test. Figure C-6 is the screenshot of the VI that computes the N2F model.

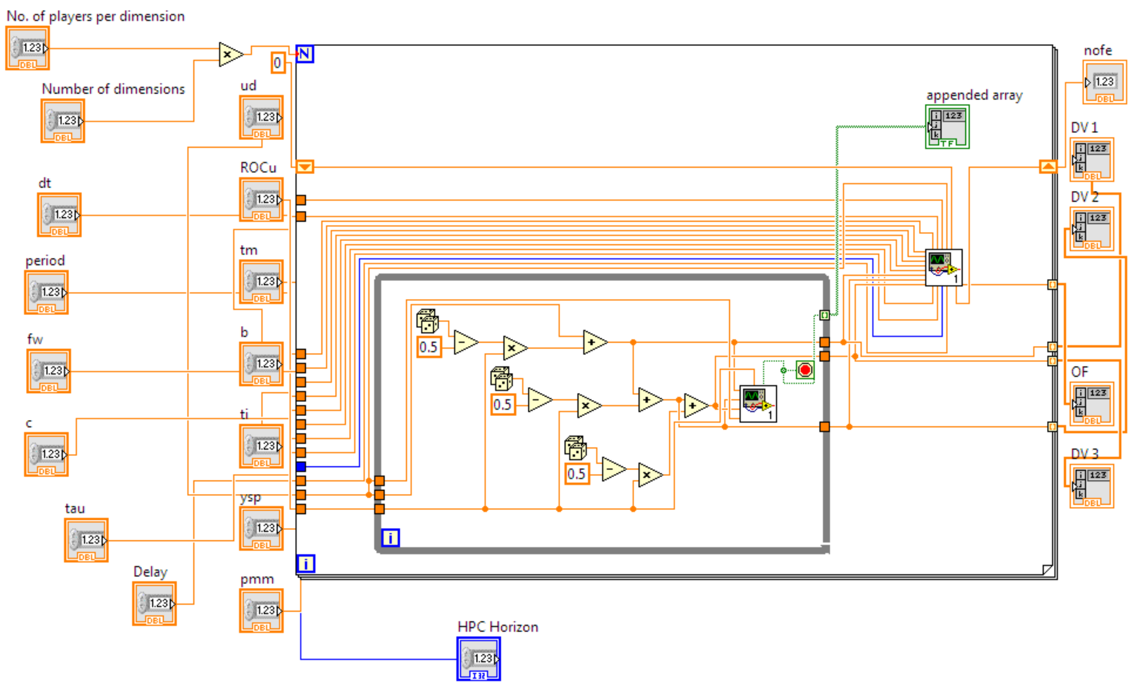


Figure C-4: Screenshot of Player Initialization

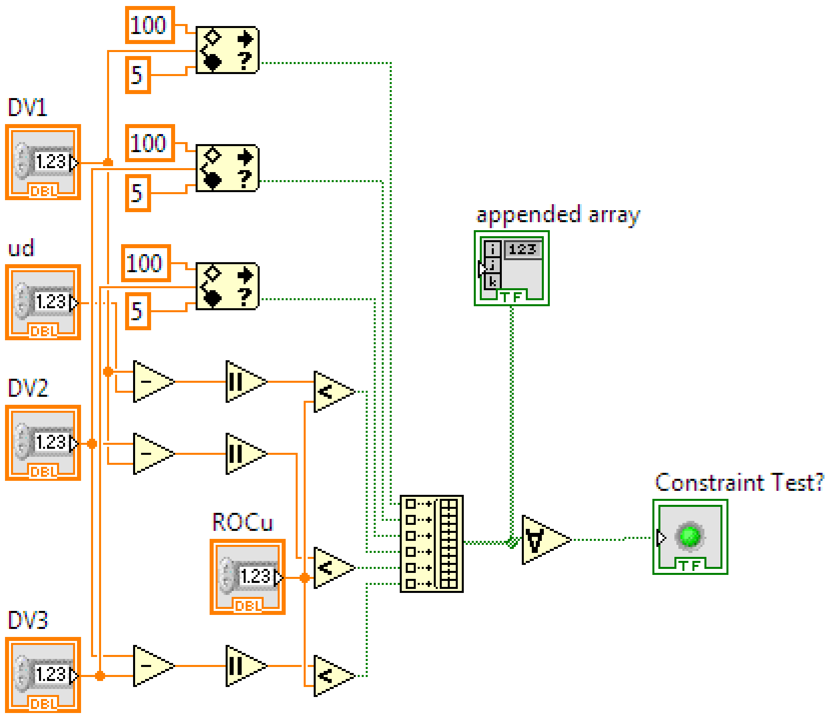
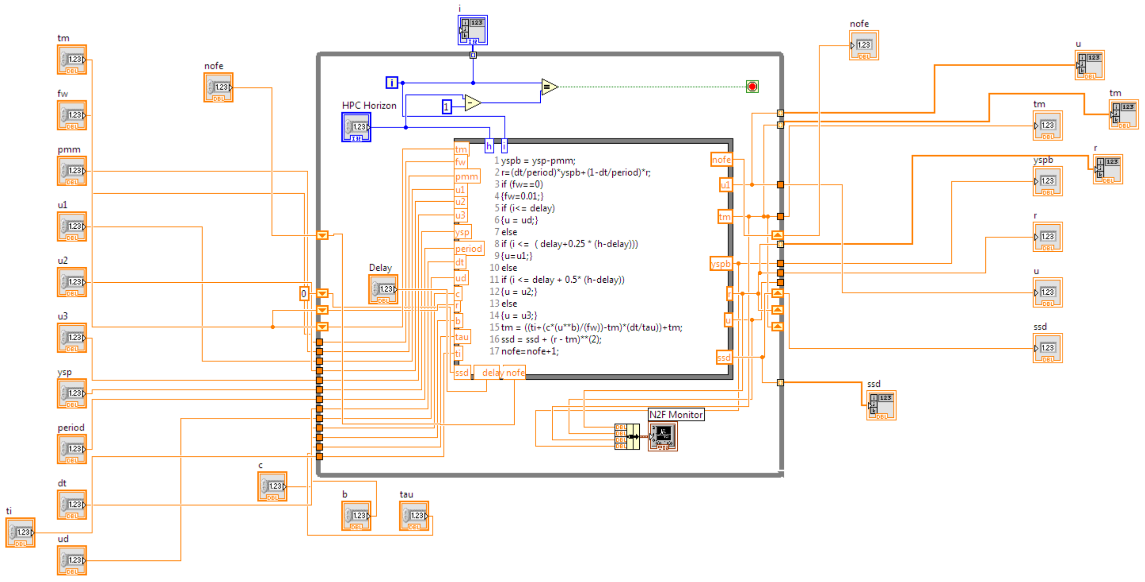


Figure C-5: Screenshot of Rate of Change And MV Constraint



```
yspb = ysp-pmm;
r=(dt/period)*yspb+(1-dt/period)*r;
if (fw==0)
{fw=0.01;}
if (i<= delay)
{u = ud;}
else
if (i <=  ( delay+0.25 * (h-delay)))
{u=u1;}
else
if (i <= delay + 0.5* (h-delay))
{u = u2;}
else
{u = u3;}
tm = ((ti+(c*(u**b)/(fw))-tm)*(dt/tau))+tm;
ssd = ssd + (r - tm)**(2);
nofe=nofe+1;
```

Figure C-6: Screenshot of N2F Model Computation

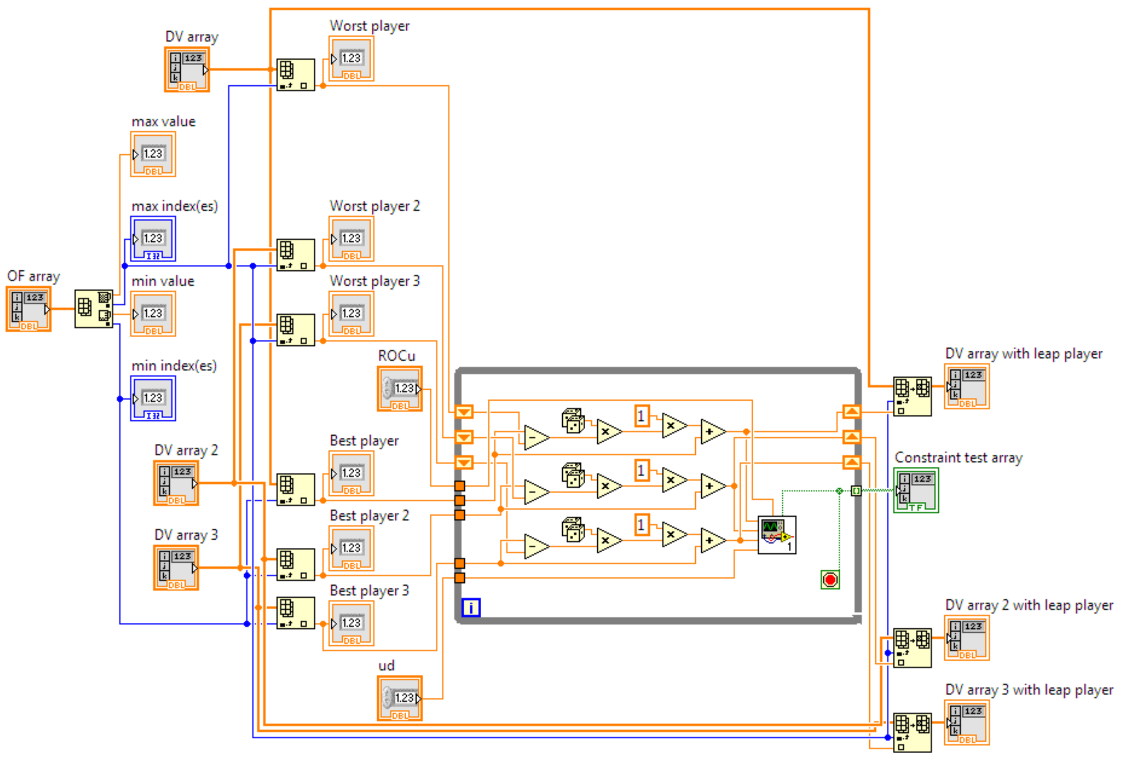


Figure C-7: Screenshot of Leap Over

In the LeapfrogOpti VI after initialization, the named VI present at the farther left within the loop is where the leap over is programmed. The best and worst player is found and then the worst player is leapt over the best player within this VI. Figure C-7 is the screenshot of the leap over VI. After every leap over N2F model is called again to compute the OF value. The VI present next to the leap over VI is the N2F model VI.

The next VI present to the right of the N2F model VI checks the stopping criteria for the Leapfrogging optimizer. Figure C-8 is the stopping criteria VI.

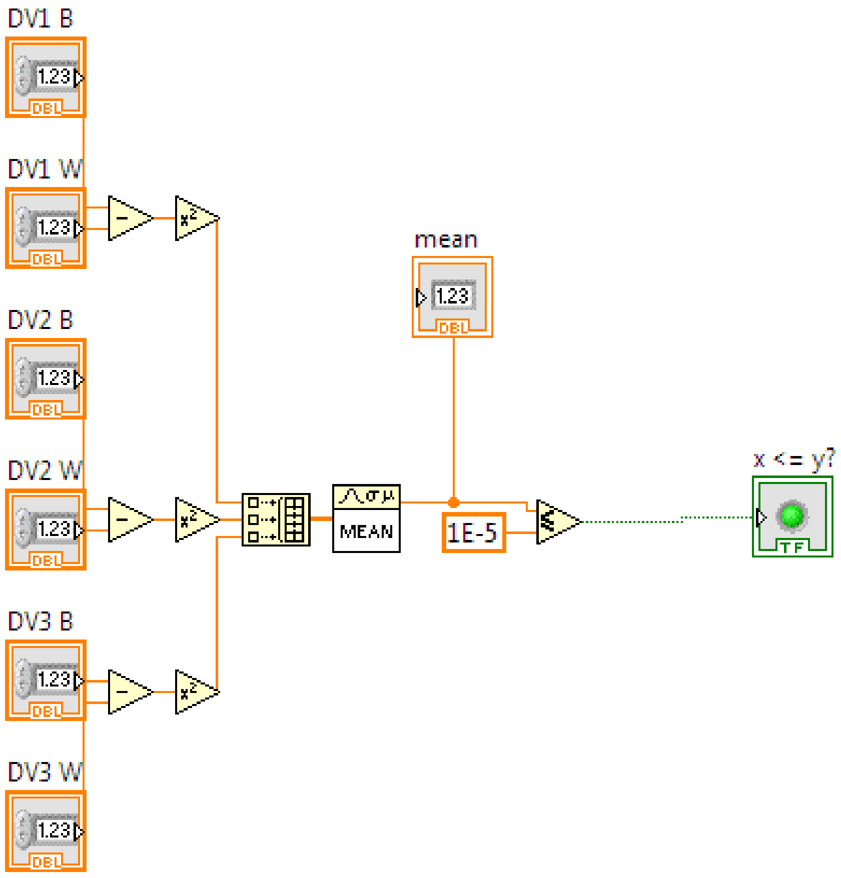


Figure C-8: Screenshot of Stopping Criteria

VITA

Upasana Manimegalai Sridhar

Candidate for the Degree of

Doctor of Philosophy

Thesis: DEVELOP AND ANALYZE THE LEAPFROGGING OPTIMIZATION TECHNIQUE: DEMONSTRATE ITS APPLICATION ON HORIZON PREDICTIVE CONTROL OF A HEAT EXCHANGER

Major Field: Chemical Engineering

Biographical:

Education:

1. Doctor of Philosophy in Chemical Engineering at Oklahoma State University, Stillwater, Oklahoma, USA in July, 2014.
2. Master of Science in Chemical Engineering at Oklahoma State University, Stillwater, Oklahoma, USA in December, 2010.
3. Bachelor of Technology in Chemical Engineering at Sri Sivasubramania Nadar College of Engineering, Kalavakkam, Tamil Nadu, India in May, 2009.

Experience:

1. Graduate Research Associate, School of Chemical Engineering, Oklahoma State University, Stillwater, Oklahoma, USA; August 2009 to July 2014
2. Graduate Teaching Associate, School of Chemical Engineering, Oklahoma State University, Stillwater, Oklahoma, USA; August 2009 to May 2014
3. Student Research Assistant, Indira Gandhi Center for Atomic Research, Kalpakkam, Tamil Nadu, India; December 2008 to May 2009

Professional Memberships:

International Society of Automation
Omega Chi Epsilon Honor Society, OSU
American Institute of Chemical Engineers
Chemical Engineering Graduate Student Association