Optimization of Heat Exchanger in Vuilleumier Heat Pump Using Teaching-Learning Based Optimization (TLBO) Algorithm

A Thesis Presented

by

Siddhartha Gadiraju

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Master of Science

in

Mechanical Engineering

Stony Brook University

May 2014

Stony Brook University

The Graduate School

Siddhartha Gadiraju

We, the thesis committee for the above candidate for the

Master of Science degree, hereby recommend

acceptance of this thesis.

Jon P. Longtin – Thesis Advisor
Professor- Department of Mechanical Engineering

Sotirios Mamalis – Chairperson of Defense
Assistant Professor- Department of Mechanical Engineering

Lin-Shu Wang- Committee Member
Associate Professor- Department of Mechanical Engineering

This thesis is accepted by the Graduate School

Charles Taber
Dean of the Graduate School

Abstract of the Thesis

Optimization of Heat Exchanger in Vuilleumier Heat Pump Using Teaching-Learning Based Algorithm

by

Siddhartha Gadiraju

Master of Science

in

Mechanical Engineering

Stony Brook University

2014

Air conditioning, space heating, and water heating makeup the largest proportion of building energy use in the U.S., making this segment an enormously rich opportunity for energy and cost savings. Devices like Vuilleumier Heat Pump (VHP), which can provide space heating, cooling and water heating at the same time, have very high efficiency compared to conventional air conditioners, space heaters and water heaters available in the market. The heat exchangers of a VHP play a significant role, and it affects the performance of a VHP significantly. Multi-Objectives such as reducing the pressure drop and dead volume of the working fluid in the heat exchangers should be considered while designing.

The primary objective of this thesis is to develop an optimization tool using Teaching-Learning based optimization (TLBO) algorithm, which considers minimizing dead volume and pressure drop as the objectives. A design process starts by assuming some random desired values for few unknowns and evaluating the rest of the parameters based on those assumptions. But, quite often, it is apparent that, as the design is refined, there might be a huge difference between initially assumed values and current design parameters. This process consumes time, effort and resources. As an alternative, implementation of optimization techniques such as TLBO over the design variables in a specified range, which can give a set of optimal solutions in very less time, is discussed in this thesis. This work shows the application of TLBO and the process of converging to an optimal solution. The results obtained by implementing TLBO for a VHP heat exchanger are then compared to the existing design. A tool like this can be used at the start of design process so that lot of time and resources can be saved in the development cycle.

Dedication

This thesis is lovingly dedicated to my friend Abhinand Palicherla. His encouragement and help has constantly supported me throughout my stay at Stony Brook University.

Table of Contents

List of Tables

Figure **Page**

Acknowledgments

I would like to thank my advisor, Professor Jon Longtin for the opportunity do my graduate studies under his guidance. His extensive knowledge in many areas of engineering has proven invaluable in allowing me to do this work.

I would like to thank Mr. Paul Schwartz, CEO of ThermoLift, Inc. and his team for support and assistance in this project.

I would like to thank Professor Sotirios Mamalis and Professor Lin-Shu Wang for being on my thesis committee. I appreciate their valuable time.

I would also like to thank Mr. Yogesh Bedekar for constantly helping and supporting me as a fellow graduate student.

Finally, I would like to thank my family members for their love and support. In spite of staying thousands of kilometers away from them, their blessing are always with me.

Chapter 1

1 Introduction

Air conditioning, space heating, and water heating make up the largest proportion of building energy use in the U.S., making this segment an enormously rich opportunity for energy and cost savings. Devices like Vuilleumier Heat Pump (VHP), which can provide space heating, cooling and water heating at the same time, have very high efficiency compared to conventional air conditioners, space heaters and water heaters available in the market. VHP also has advantages in respect to versatility of the fuel used. Being an external combustion heat pump, VHP can run on natural gas, liquid fuel or solar heating

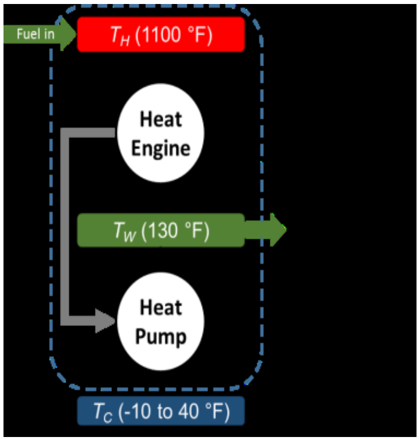

Figure 1.1 VHP Concept

The Vuilleumier heat pump (VHP) can be thought of as a heat engine directly coupled to a heat pump, as shown Figure 1.1. Fuel is burned to produce a high-temperature heat source at temperature$\mathrm{T_H}$, typically around 1,100 °F. This heat source provides heat $\mathrm{Q_H}$ to power the heat engine. Waste heat from the engine, $\mathrm{Q_W}$, is discharged to the warm-temperature reservoir at temperature $\mathrm{T_w}$, typically 130 °F, which represents the heat source for the house, e.g., the water temperature in a hydronic heating system. Work W from the heat engine is coupled directly to the heat pump, which pumps heat from the outdoor ambient at temperature$\mathrm{T_c}$, typically–10 to +40 °F. The total thermal energy delivered to the home is$\mathrm{Q_H}(1 + \eta\beta)$, where η is the heat engine efficiency, and β is the heat pump coefficient of performance. For a traditional boiler or furnace, the maximum heat that can be delivered is $\mathrm{Q_H}$, assuming a 100% efficiency. In contrast the VHP provides an increase of $\eta\beta$ in the heat delivered, where $\eta\beta$ can range from 0.6 to 1.2.

The design of heat exchangers of a VHP using TLBO algorithm is discussed in this work. The heat exchangers of VHP have to be designed to extract specific amount of heat as dictated by parameters of the VM cycle which are external to the design of heat exchangers. Unlike most heat exchangers, where minimizing the capital is the primary objective in design process, the heat exchangers of the VHP have to be designed to minimize the pressure loss and dead volume of the working fluid. These two objectives are defined in further chapters.

Traditionally, a design process starts by assuming some random values to the unknown parameters based on designers' experience. But, when there are multiple parameters which not only affect the heat exchanger performance but also affects the efficiency of the entire VHP, a random guess while assuming values to some parameters at the start of the design process may lead to more time and resources spent on validation and refinement of initial design. In this process, even if a designer was able to design an efficient heat exchanger, it may not guarantee to be an ideal heat exchanger for a VHP. So, a comprehensive optimization problem which takes into consideration multiple objectives such as pressure drop and dead volume is needed, while also satisfying the constraints of handling the required amount of heat transfer. The solution obtained from the optimization problem may provide a good start and direction for further design refinement using experimental or computational techniques.

1.1 Brief History of Vuilleumier Heat pump and benefits

The earliest description of the Vuilleumier Heat Pump was given by the Rudolph Vuilleumier in 1918 in a US Patent (US Patent No. 1275507, 1918). Early research on VHP has been reported at Massachusetts Institute of Technology by Professor Vannevar Bush in 1938 (US Patent No. 2127286, 1938) (US Patent No. 2157229, 1939). Between 1968 and 1979 Hughes Aircraft Co., Culver city, California and Phillips Laboratories, Briarcliff Manor, NY has done significant work on VHP which claimed high reliability and potential for long life (Magee, 1968). Further, a double expansion Vuilleumier cycle engine for low temperature infrared applications has been developed at Hughes (Doody, 1971) . Also, an ultra-miniature split Vuilleumier cooling engine has been designed and developed at Hughes (Berry, 1973). A study of Vuilleumier cycle cryogenic refrigerator for detector cooling on the limb scanning infrared radiometer submitted (Russo, 1976) for NASA describes briefly most of the work done at the Hughes on VHP. The report and work done by Ronald White described the design and operating parameters of the engine and it serves as the best resource for understanding a VHP (White, 1976) (White, 1973) (White, 1976). Around the same time Philips Laboratories developed miniature engine of VHP for the Night Vision Laboratory, Fort Belvoi, Maryland (Daniels, 1971) (Pitcher G. , 1970) (Pitcher G. , 1973) (Pitcher G. d., 1970). AiReserch Manufacturing CO., Torrance, California worked for NASA-Goddard Space Flight Center (GSFC), Greenbelt, Maryland for the comprehensive analysis and design optimization for a split Vuilleumier Cooling Engine to be operated at ambient environment (Cohen, 1973). But further work was abandoned in favor of Sterling cooling engines.

In Europe, a gas fired residential heater working on Vuilleumier Thermodynamic Cycle has been demonstrated (Carlsen, 1989). Between 1994 and 1999 BE Thermolift GbR, Aachen, Germany in coordination with University of Dartmund, Germany worked to design, develop and test the VHP (Schlutz, 1995). The report demonstrated a VHP fulfilling the requirements for the role of a heat generator for residential heater with domestic water heating in conjunction with a hot water heating pump. The two continuous runs, with a heating capacity of 33kW and a 4kW heat pump, had by the end of the project reached respective runs times of more than 6,000 operating hours; the cumulative run time of the heat pump added to the run time of the functional model resulted in a total experimental result of more than 20,000 operational hours.

1.1.1 Advantages:

- There are very few moving parts and it is a closed cycle with Hydrogen or Helium as working fluids. It can be hermetically sealed.
- A VHP can be fuel-agnostic. Theoretically any heat source which can reach the desired parameters can act as energy source.
- Both heating and cooling at the same time, thus saving a huge margin in energy costs.
- A VHP can be run grid independent and can provide heating and cooling even during peak load periods with minimized cost.
- Low mechanical forces on moving parts give an increased life.
- Significant weight reduction due to careful selection of materials for the displacers.

1.2 Motivation

A typical VHP has three heat exchangers. The efficiency of the VHP is highly dependent on the efficiency of the heat exchangers. The design of heat exchangers for a VHP involves lot of unknown variables which in most cases are external to the operation of Heat Exchanger. This makes it difficult to design. In the initial process of the design, values for few variables have been assumed and the design has been evaluated for the remaining variables. This process gave a design which looked promising for an efficient heat exchanger. But, reducing the dead volume and pressure drop were not the main criteria in the process, rather obtaining the sufficient heat transfer capacity has been the important factor. Before a final design was considered, lot of analytical calculations and computational analysis were performed which involved lot of time and use of resources. It has been observed that, there is a huge difference between the initial assumed values and the final design. The huge difference can be attributed to the random guess made by the decision maker during initial design.

This has been the motivation point. This work is a result of a quest in finding a scientific technique for estimating the design variables. Ideally, an optimized design for a VHP heat exchanger is one which can minimize the dead volume and pressure drop. There is a need for implementation of design optimization process which can generate a better set of solutions to initiate the design process rather than some random guess. After an extensive literature survey about optimization methods, the Teaching-Leaning based optimization (TLBO) algorithm has been short listed. TLBO is nature inspired evolutionary algorithm which is based on the natural process of teaching-learning which happens commonly all around.

1.3 Objective and thesis outline

The primary objective of this work is to show the working process of a TLBO algorithm and compare how the results are converged to a set of optimal solutions called Pareto Optimal solutions. The solutions obtained by the implementation of the TLBO are compared to the existing design. A function generated by considering both dead volume and pressure drop, while satisfying various constraints of heat exchangers acts as the comparison criteria or fitness value. This function at different phases of the algorithm gives a fitness value for the solution set at that phase.

Chapter two outlines the basic operation and thermodynamic cycle of the VHP. It also discusses the heat exchanger theory which is applicable to the heat exchangers of a VHP. Dead volume and pressure drop are discussed as well. The concept of optimization, various optimization techniques and TLBO has been discussed in chapter three. Multi-objective optimization techniques have been listed and the Linearization technique which has been used in the present work has been discussed. Constraint handling has also been discussed.

Chapter four show some specification of the existing design which later is compared with the solutions obtained from TLBO implementation. TLBO algorithm initialization, various phases and the process of solution being converges is illustrated in chapter five. Chapter Six shows various results and a comparison study with the existing design. Chapter seven gives the conclusion and scope for future work.

Chapter 2

2 Vuilleumier Heat Pump

2.1 Theory of Operation

The Vuilleumier refrigeration cycle is invented and patented by Rudolph Vuilleumier in 1918 (US Patent No. 1275507, 1918). In most refrigeration and heat pump cycles, the required energy is supplied as mechanical energy by means of motor. But for a VHP energy supply is by means of a heat source which typically is at 1100°F. The Vuilleumier (V-M) cycle absorbs heat at both high and low temperatures and rejects this heat at some intermediate temperature. The heat input, mostly by natural gas burning, at the hot end provides required energy for cooling effect at the cold end. While the heat absorbed at both cold and hot end is rejected at the ambient end called warm end.

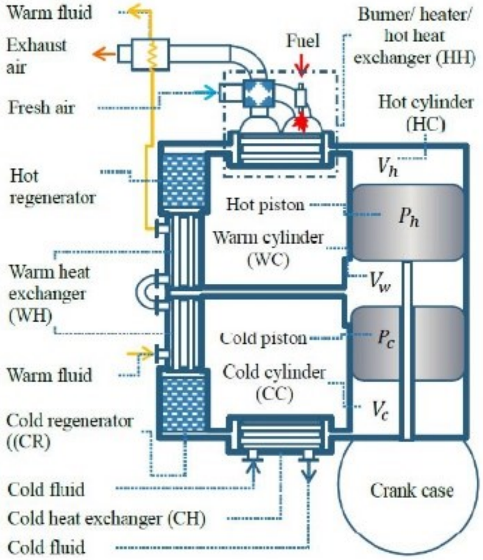

Figure 2.1 VHP Concept II

Basically a machine operating on V-M cycle has three chambers separated by two displacers (P_h, P_c). The three chambers are hot, warm (ambient) and cold chambers which are represented by V_h, V_w, V_c in the Figure 2.1.

2.2 Thermodynamic Cycle

As described earlier the Vuilleumier heat pump primarily comprises of three volumes, hot (Vh), Warm (Vw) and Cold (Vc). As the displacers move to and fro in their axis with 90° phase angle, the gas in the respective chambers are pushed out. Typically there are no valves in a VHP. Hence, as the displacers move to one end of a chamber, the gas in that chamber is pushed out freely. Also, since there are no valves, the working fluid is always at the same pressure at any given point of time across the entire VHP.

When the hot displacer move towards its bottom dead center, the hot active volume increases and since the hot end is supplied with constant source of energy, the temperature of the gas increases and thus the pressure of the gas also increases. As discussed earlier, since there are no valves in a VHP, the pressure of the working fluid is increased everywhere in the VHP. On the other hand, as the hot displacer travels to its top dead center, the hot gas is pushed out of the hot volume and into the warm volume through hot regenerator. Regenerator is mass of metal, porous enough to let the gas flow through it. As the hot gas flows through the regenerator, it deposits its energy to the metal and thus the temperature of the gas is significantly reduced. Before the gas occupies the warm chamber, the gas pass through a warm heat exchanger and temperature of the gas is further reduced and thereby decreasing the pressure throughout the system. The motion of the hot displacer in a VHP, thus give a pressure variation in the system as a function of displacer movement. Thus it acts as a thermal compressor.

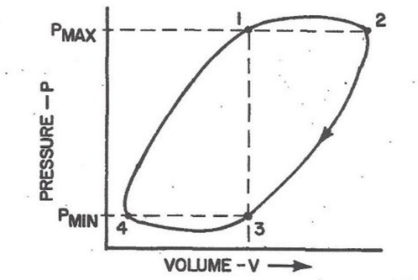

Figure 2.4 P-V Chart for Hot Volume

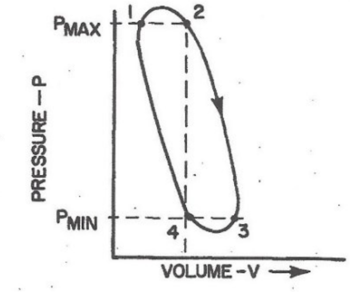

Figure 2.3 P-V Chart for Cold Volume

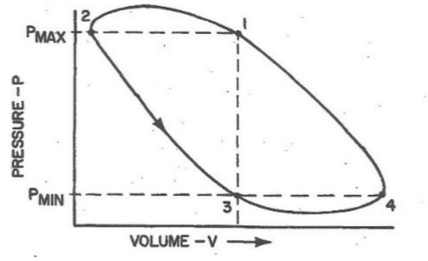

Figure 2.2 P-V Chart for Warm Volume

The cold displacer, simultaneously, with 90° phase angle compared to hot displacer moves to-and-for displacing the gas in cold chamber in to warm chamber and vice versa. As the pressure in the system decreases, simultaneously, the cold displacer move towards the the warm chamber and subsequently, the cold chamber is filled with gas. In this process as the gas move from the warm chamber to the cold chamber, it passes through the cold regenerator which takes out more energy from the gas and thus reducing the temperature of the gas. Since this process simultaneously occurs with the decrease of pressure in the system due to hot displacer motion, the pressure in the cold volume is also decreased and thus the temperature is further reduced.

If this cold end is paired with a heat sink which is at higher temperature than the cold end, then there is a transfer of energy into the system from the heat sink.

The gas is then pushed out of the cold chamber as the cold displacer moves to its bottom dead center. The gas again pass through the cold regenerator and since the gas is at lower temperature than the regenerator, the gas absorbs more energy, which it dissipates while passing through the warm heat exchanger and subsequently occupying the warm chamber. The cycle repeats itself, thus producing heating effect at warm end and cooling effect at the cold end at the same time.

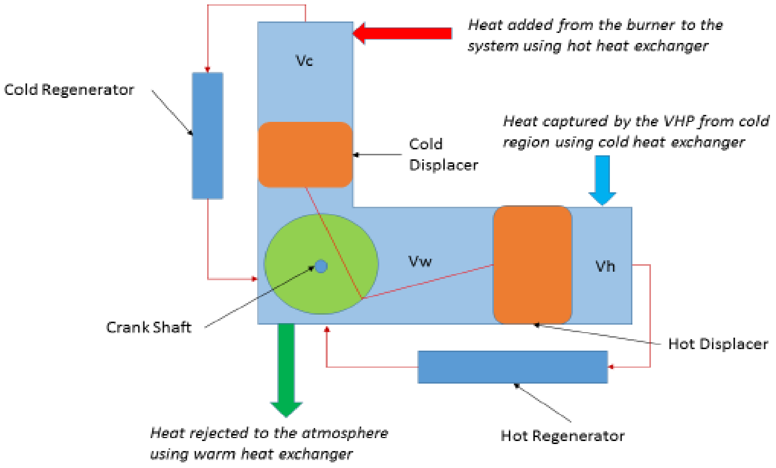

Figure 2.5VHP Operation-Block Diagram

2.3 Components

The primary components of the VHP are heat exchangers, regenerators, displacers and driving mechanism. There are three heat exchangers, two regenerator and two displacers. Since the primary objective of this work is design process of heat exchangers, the heat exchanger design and parameters are discussed in the following sections.

2.3.1 Heat Exchangers and its equations

Warm heat exchanger design for a VHP are considered in the present study. The warm heat exchanger in a VHP is similar to classic Shell and tube heat exchangers. Helium or working gas flows in the tubes and water, which acts as a coolant, flow in the shell side of the heat exchanger.

2.3.1.1 Heat transfer aspects

The starting point of any heat transfer calculation is the overall energy balance and heat transfer rate equation. Since there is no phase change in the VHP, only sensible heat is transferred. We can write heat Q as

Equation 2.1 $Q = \dot{m}_{He}\, C_{p,He}\, (T_{He,in} - T_{He,out}) = \dot{m}_{water}\, C_{p,water}\, (T_{water,out} - T_{water,in})$

Equation 2.2 $Q = U\, A\, \Delta T_{lm}$

The convention for ΔT_{lm} in shell-and-tube heat exchangers is as follows:

Equation 2.3 $\Delta T_{lm} = \frac{(T_{He,in} - T_{Water,out}) - (T_{He,out} - T_{Water,in})}{ln(\frac{T_{He,in} - T_{Water,out}}{T_{He,out} - T_{Water,in}})}$

(Heat exchanger schematics)

The Overall heat transfer coefficient of the heat exchanger can be calculated as described below

Equation 2.4 $\frac{1}{U_o} = \frac{1}{h_o} + \frac{\Delta r}{k} \frac{A_o}{A_{lm}} + \frac{1}{h_i} \frac{A_o}{A_i}$

The heat transfer on the tube side can be calculated in a standard approach. Nusselt number equation (Holman, 2002) for a duct flow in turbulent regime which holds for 2500< $Re_D < 10^6$ and Pr<120 is given as

Equation 2.5 $Nu_{he} = 0.023\, Re_{D_i}^{4/5}\, Pr^{0.3}$

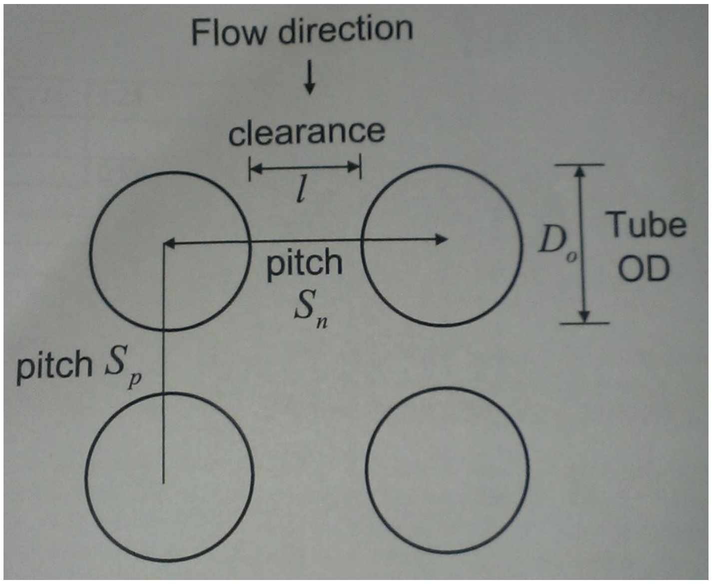

Figure 2.6Clearance and Pitch for Bundled tubes

All the physical properties of the fluid are evaluated at bulk temperature.

On the shell side, correlation for the flow through tube banks are used, as given in the book by (Holman, 2002).

Equation 2.6 $Nu_{h2o} = C\, Re_{D_o}^{n}\, Pr^{1/3}$

Where $Re_{D_o} = \frac{D_o V_{max} \rho}{\mu}$

D_o is the outside diameter; V_{max} is maximum velocity of the fluid through the tube bank. Segmented baffles are considered in the present study. While evaluating the Reynolds numbers on the shell side, cross flow velocity across the tube bundle has to be determined. For the length scale, the tube outside diameter can be employed. To find the maximum velocity, cross-flow area must be evaluated.

Equation 2.7 $Cross\ flow\ area = Shell\ ID\ \times Baffle\ spacing\ \times \frac{Clearance}{pitch}$

Where clearance l and pitch Sn are illustrated in the Figure 2.6. The clearance $l = S_n - D_o$. All the properties of the fluid are evaluated at the arithmetic average temperature of the fluid between the two end temperatures.

S_p/D_o S_n/D_o	**1.25**	**1.5**	**2.0**	**3.0**
1.25	0.386	0.305	0.111	0.0703
1.5	0.407	0.278	0.112	0.0753
2.0	0.464	0.322	0.254	0.220
3.0	0.322	0.396	0.415	0.317

Table 2.1 Values for constant C

S_p/D_o S_n/D_o	**1.25**	**1.5**	**2.0**	**3.0**
1.25	0.592	0.608	0.704	0.752
1.5	0.586	0.620	0.702	0.744
2.0	0.570	0.602	0.632	0.648
3.0	0.601	0.584	0.581	0.608

Table 2.2 Values for constant n

2.3.1.2 Pressure loss and Dead Volume

Vuilleumier heat pumps are highly sensitive to the flow restrictions. COP values of these heat pumps are significantly affected by the pressure drops in the working gas as it moves between different chambers of the machine. One of the other parameters which affects the performance of the VHP is the dead volume. Pressure drop and the dead volume are inversely related. The heat exchangers need to be designed appropriately so as to have minimal pressure losses while maintaining lower dead volumes. Since the helium flow is in the tube side, tube side pressure loss can be calculated as

Equation 2.8 $\Delta P = f\ \frac{L}{D_i}(\frac{1}{2}\rho\, V^2)$

Dead volume is the volume in a VHP where the Helium or working gas can flow. This is the volume which working fluid can occupy excluding the swept volume of the displacers. In respect to the heat exchanger in consideration, the dead volume is the inside volume of all the tubes.

Equation 2.9 $V_{dead} = n\,\pi\ \frac{{D_i}^2}{4}\ L$

Chapter 3

3 Optimization Techniques

3.1 Basic concept

Optimization techniques are a collection of mathematical results and numerical methods for identifying the optimal solution from a set of possible solutions without explicitly having to evaluate all possible alternatives. The root of engineering lies in designing new, better, and less expensive design alternatives as well as to devise plans and procedures for improvement of the existing design. The power of finding an optimal solution without enumerating all possible alternative comes from the use of a modest level of mathematics and iterative numerical methods. This involves defined methods and algorithms implemented on computing platforms..

3.2 Multi Objective optimization

Multi-objective optimization is concerned with decision making and optimization problems where there is a need to optimize more than one function or objective simultaneously. Most of the engineering design and optimal criteria demands a tradeoff between multiple criteria or objectives. In most of the multi-objective problems, there does not exist one particular solution that satisfies all the Objectives simultaneously. Such cases can be seen as multiple objectives conflicting each other and there will be a set of solutions called Pareto optimal solutions (Multi-objective optimization - Wikipedia, the free encyclopedia). A solution is called Pareto optimal if none of the objective functions can be improved in value without degrading the others. Mathematically multi-objective optimization can be seen as

Equation 3.1 $$min\left(f_1(x), f_2(x), \ldots \ldots \ldots . f_n(x)\right) \quad | \, x \in X$$

Where $k >= 2$ and set X is feasible set of decision vectors.

On a broader classification, solving a multi-objective optimization problem can be classified into four ways,

I. Scalarizing
II. No-preference methods

III. Priori methods
IV. Posteriori methods.

The method of Scalarizing has been used in this work. Scalarizing is essentially formulating a single objective problem by scalarizing multi-objectives, such that the solution acts as Pareto optimal solution for multi-objective optimization. Scalarization parameters are used to scalarize a multi-objective problem into single optimization problem. With different value to the scalarization parameters, different Pareto optimal solution are obtained. The most general formulization for scalarization is

Equation 3.2 $\min_{x \in X} \sum_{i=1}^{k} w_i \, f_i(x)$,

Where w_i is the scalrization factor which is always greater than zero.

3.3 Constraint Handling

Almost all the engineering optimization problems have constraints. All the optimal solutions obtained by the optimization problem must necessarily be feasible and satisfy all constraints. As the number of constraints increase, difficulty in handling them increases. In most of the cases, an increase in difficulty is more exponential rather than linear. One important aspect while dealing with constraint handling is its criticality, in terms of absolute satisfactions. If small violations can be considered for the final solution for otherwise superior solution, it is distinguished as a soft constraint handling. Evolutionary algorithm, like the one considered in this thesis, are more suitable for such constraints.

Penalty functions are most commonly used constraint handling methods. Penalty function method can be seen as a method that does not allow the temporary solution in the evolutionary process to stray too far from the feasible solution by modifying the objective function. For a given objective function such as

Equation 3.3 $\min f(x) \;\; | \, x \; \in A$

$x \; \in A$ is the constraint in this problem. Penalty methods, constraint handling for such problem can be reformulated as

Equation 3.4 $min\, f(x) + p(d(x, A))$

Where $d(x, A)$ is metric function which evaluates the distance between solution vector$x$, and the region A. While $p(.)$ is monotonically non decreasing function with $p(0) = 0$. Various functions for these both functions gives variety of constraint handling techniques. (Thomas Baeck, 1995). Static penalty method has been used in this work. A general formulation for a minimization problem is given as

Equation 3.5 $$f_p(x) = f(x) + \sum_{i=1}^{m} C_i d_i^k \quad Where\ d_i = \begin{cases} \delta_i\, g_i(x), for\ i = 1, \dots, q \\ |h_i(x)|,\ for\ i = q + 1, \dots, m \end{cases}$$

Usually k is taken either 1 or 2. q is the total number of inequality constraints and m is the total number of constraints including equality constraints. δ_i is extremely small number while C_i is a large number which avoids the solution not to stray away from feasible solution too much.

Other techniques include dynamic penalty approach, adaptive penalty approach and other such functions. In almost all the penalty function approaches, the main difficulty is finding the number C_i. There is no direct methods for finding it. As the number of constraints increase it becomes more difficult. Fortunately in this thesis, only one constraint has been considered.

Various Optimization techniques

Optimization techniques can be classified as shown in the following figure.

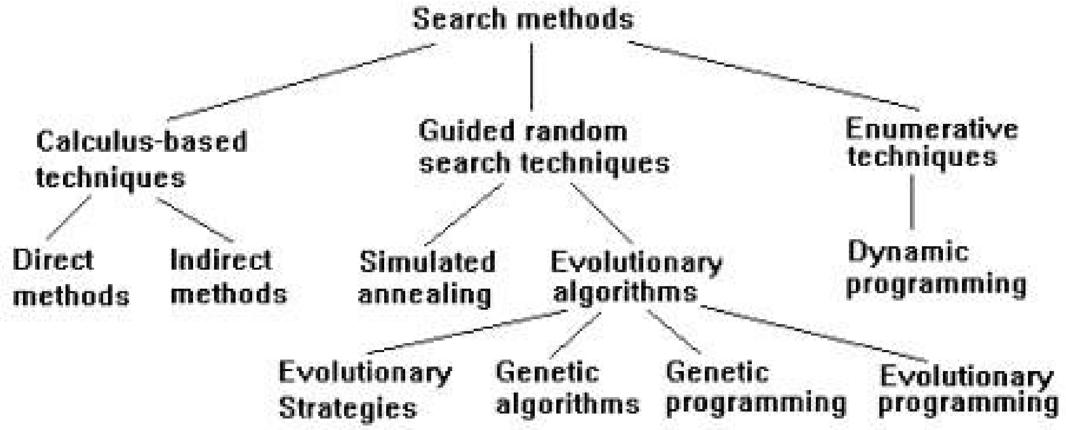

Figure 3.1 Various Optimization Techniques

3.3.1 Calculus based Techniques:

Calculus based methods of optimization are basic forms of non-linear programming. Basic techniques for calculus based optimization are as follows

- Maxima-minima (both local and global)
- The Hessian function
- Second derivative test
- Lagrange Multipliers
- Karush-Kuhn-Tucker condition
- Lagrange multipliers on Banach spaces

3.3.2 Enumerative Techniques:

This techniques involve solving the combinatorial problems. Dynamic programming is the most extensively used enumerative technique. Dynamic Programming involves solving complex problems by breaking them down to small problems. This algorithm enumerates all possible ways to solve the problem and will pick up the best solution. Dynamic programming can be roughly described as an intelligent, brute force method to pick up the best solution in less time. (Dynamic Programming)

3.3.3 Guided Random Search Techniques:

There are many guided random search algorithms, such as Genetic Algorithm (GA), Particle Swarm Optimization, Ant-Colony optimization, Simulation Annealing method, TLBO, etc., There methods are indirect way of optimization and most of them are nature inspired and evolutionary type methods.

3.4 Teaching-Learning Based Algorithm

Most of the description about TLBO has been extensively referenced from (R Venkata Rao, 2012) and (Sites). All the evolutionary and swarm intelligence based algorithms require common controlling parameters like population size and number of generations. Apart from these common control parameters, different algorithms require their own algorithm-specific control parameters. For example, GA uses mutation rate and crossover rate; PSO uses inertia weight, social and cognitive parameters; ABC uses number of bees (employed, scout and onlookers) and limit; HS requires harmony memory consideration rate, pitch adjusting rate and the number of improvisations; ACO requires exponent parameters, pheromone evaporation rate and reward factor; etc. Sometimes, the difficulty in the selection of algorithm-specific control parameters increases with modifications and hybridization. The proper tuning of the algorithm-specific parameters is a very crucial factor which affects the performance of the optimization algorithms.

The improper tuning of algorithm-specific parameters either increases the computational effort or yields the local optimal solution. Considering this aspect, the Teaching-Learning-Based Optimization (TLBO) algorithm does not require any algorithm-specific control parameters. TLBO requires only common controlling parameters like population size and number of generations (and elite size, if considered) for its working. Thus, TLBO can be said as an algorithm-specific parameter-less algorithm.

TLBO is a teaching-learning process inspired algorithm proposed by Rao et al. (R Venkata Rao, 2012) based on the effect of influence of a teacher on the output of learners in a class. The algorithm describes two basic modes of the learning: (i) through teacher (known as teacher phase) and (ii) interacting with the other learners (known as learner phase). In this optimization algorithm a group of learners is considered as population and different subjects offered to the learners are considered as different design variables of the optimization problem and a learner's result is analogous to the 'fitness' value of the optimization problem. The best solution in the entire population is considered as the teacher. The design variables are actually the parameters involved in the objective function of the given optimization problem and the best solution is the best value of the objective function. The working of TLBO is divided into two parts, 'Teacher phase' and 'Learners phase'.

3.4.1 Teacher phase:

During this phase a teacher tries to increase the mean result of the class in the subject taught by him or her depending on his or her capability. At any iteration i, assume that there are m number of subjects (i.e. design variables), n number of learners (i.e. population size, $k = 1,2, \dots, n$) and $M_{j,i}$ be the mean result of the learners in a particular subject 'j' ($j = 1,2, \dots, m$). The best overall result $X_{\mathrm{total-kbest,i}}$ considering all the subjects together obtained in the entire population of learners can be considered as the result of best learner k_{best}. However, as the teacher is usually considered as a highly learned person who trains learners so that they can have better results, the best learner identified is considered by the algorithm as the teacher. The difference between the existing mean result of each subject and the corresponding result of the teacher for each subject is given by,

Equation 3.6 $Difference\_Mean_{j,k,i} = r_i \, (X_{j,,kbest,i} - T_f \, M_{j,i}),$

Where, $X_{j,kbest,i}$ is the result of the best learner (i.e. teacher) in subject j. T_f is the teaching factor which decides the value of mean to be changed, and ri is the random number in the range [0, 1]. Value of T_f can be either 1 or 2.

T_f is not a parameter of the TLBO algorithm. The value of T_f is not given as an input to the algorithm and its value is randomly decided by the algorithm using Eq. (2). After conducting a number of

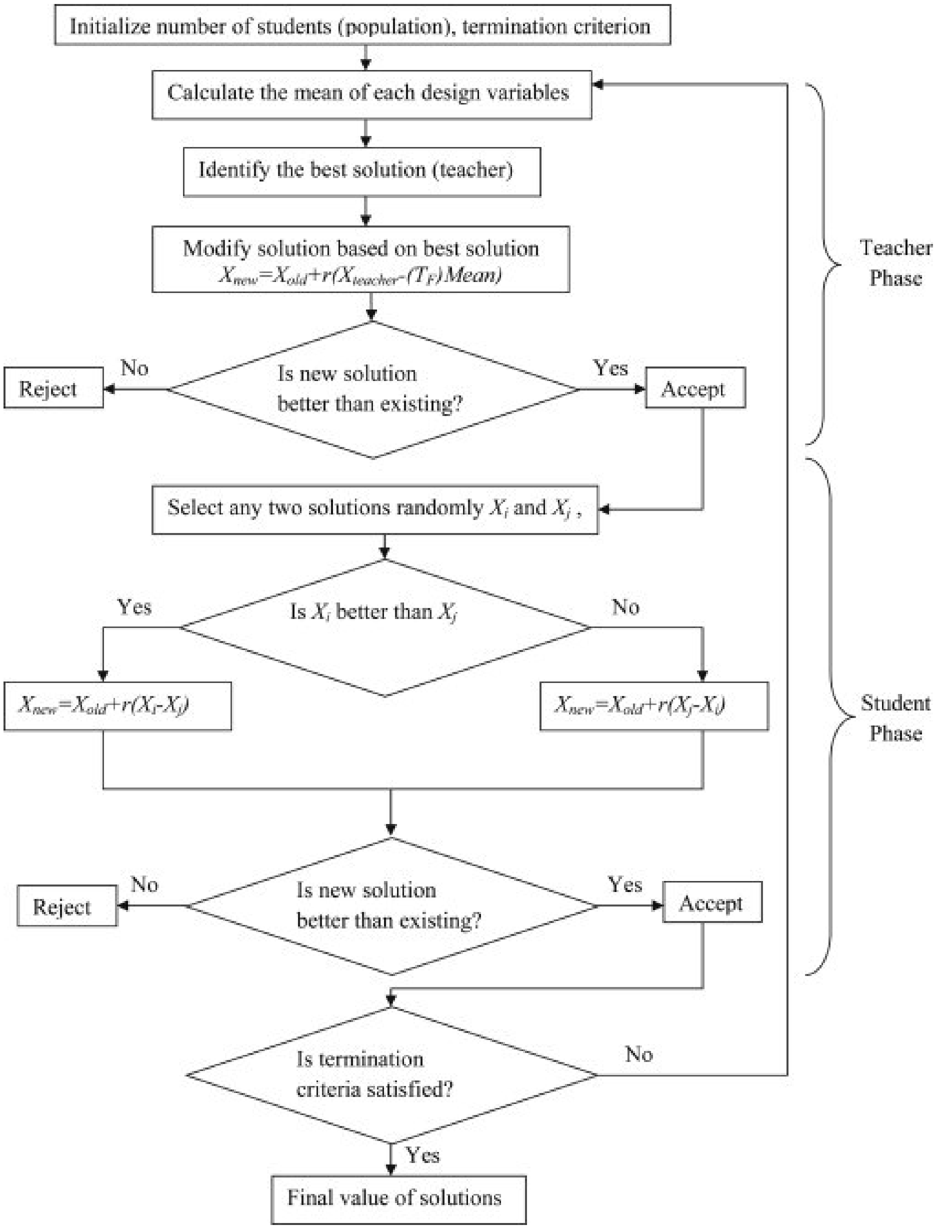

Figure 3.2TLBO Algorithm flow chart

experiments on many benchmark functions it is concluded that the algorithm performs better if the value of T_f is between 1 and 2. However, the algorithm is found to perform much better if the value of T_f is either 1 or 2 and hence to simplify the algorithm, the teaching factor is suggested to take either 1 or 2. Based on the$Difference_Mean_{j,k,i}$ the existing solution is updated in the teacher phase according to the following expression.

Equation 3.7 $X_{j,k,i}^{`} = X_{j,k,i} + Difference\_Mean_{j,k,i}$

Where $X_{j,k,i}^{`}$ is the updated value of $X_{j,k,i}$. Accept$X_{j,k,i}^{`}$ if it gives better function value. All the accepted function values at the end of the teacher phase are maintained and these values become the input to the learner phase. The learner phase depends upon the teacher phase.

3.4.2 Learner phase:

Learners increase their knowledge by interaction among themselves. A learner interacts randomly with other learners for enhancing his or her knowledge. A learner learns new things if the other learner has more knowledge than him or her. Considering a population size of 'n', the learning phenomenon of this phase is expressed below.

Randomly select two learners P and Q such that $X_{total-P,i}^{`} \neq X_{total-Q,i}^{1}$ (where, $X_{total-P,i}^{`}$ and $X_{total-Q,i}^{1}$ are the updated values of$X_{total-P,i}^{`}$ and $X_{total-Q,i}$respectively at the end of teacher phase)

Equation 3.8 $X_{total-P,i}^{``} = X_{total-P,i}^{`} + r_i \left(X_{total-P,i}^{`} - X_{total-Q,i}^{1}\right),\ If\ X_{total-P,i}^{`} < X_{total-Q,i}^{1}$

Equation 3.9 $X_{total-P,i}^{``} = X_{total-P,i}^{`} + r_i \left(X_{total-Q,i}^{`} - X_{total-P,i}^{1}\right),\ If\ X_{total-P,i}^{`} > X_{total-Q,i}^{1}$

Accept $X_{j,P,i}^{``}$ if it gives a better function value.

Chapter 4

4 Existing Design

4.1 Heat Exchanger configuration

The heat exchanger that is being considered which is a part of a Vuilleumier Heat pump is classic example of shell and tube heat exchanger. This heat exchanger is warm heat exchanger which is designed for 24kW.

The following are the specification of the heat exchanger:

Parameter	Value	Unit
Inner Diameter	0.00472	Meters
Thickness	0.0009	Meters
Number of tubes	372	Numbers
Length of the tubes	0.122	Meters
Effective Diameter of the Shell	0.144	Meters
Clearance between the tubes	0.00065	Meters
Pitch between the tubes	0.00065	Meters
Number of Baffles	3	Number
Mass flow rate of water	0.59	kg/sec

Table 4.1 Parameters for Existing Design

4.2 Heat Exchanger effectiveness

Since the objective of this optimization technique is to minimize the objective function which is a function of dead volume and pressure loss, the heat exchanger effectiveness in this regard is enumerated in terms of Objective function.

The weighted objective function, which is discussed in the next chapter, is gives as

Equation 4.1 $f = w\, \Delta P + (1-w)\, V_{dead}$

Using this Equation 4.1 and the equations Equation 2.8 and Equation 2.9 for pressure loss and dead volume respectively, the following are the results obtained for existing design.

Function	value
Pressure Loss ΔP	45.04 Pa
Dead Volume V_dead	0.000819 m^3

Table 4.2 values of Dead volume and Pressure Drop

Chapter 5

5 Algorithm Development

5.1 Weighing objective

The present problem deals with two objectives as in minimizing pressure drop in working fluid and dead volume. So linearization of the two objective is done using scalarization technique. A weight parameter w is used as given below.

Equation 5.1 $f = w\,\Delta P\ + (1 - w)V_{dead}$

Where w lies between 0 and 1. In this case the weighing parameter is takes as 0.5, which means that equal preference is given to both the objectives.

5.2 Constraint Handling

The primary constraint in this problem is the basic heat transfer equation. $Q = U\,A_{ln}\,(LMTD) = 24kW$. In other way the primary constraint is

Equation 5.2- $Q - \ 24\ kW\ = 0$

U is the overall heat transfer coefficient, which can be calculated using the equating given in Equation 2.4

5.3 Objective Function Normalization

Since dead volume, pressure drop and heat transfer have difference units and also since there is difference in magnitudes, the objective function has to be normalized. For normalization, average pressure drop and average dead volume are used as reference.

Normalized objective function is finally given as:

Equation 5.3 $f = w\,\frac{\Delta P}{\Delta P_{avg}} + (1 - w)\frac{V_{dead}}{V_{dead_{avg}}} + C\,|1 - \frac{Q}{24000}|^{1}$

5.4 TLBO for VHP Heat Exchanger

The primary objective while optimizing a VHP Heat exchanger is minimizing pressure drop and dead volume of the working fluid. The design requires to transfer 24kW of heat from working fluid to the coolant fluid. Water is the coolant in this case. Since the VHP is mostly used for the HVAC purpose, there is standard range for the input and output temperatures of the coolant. The inlet temperature of the working fluid is known and the outlet temperature can be easily calculated from the known parameters such as mass flow rate of working fluid and the required heat transfer rate. The unknown parameters in the heat exchanger design are

I. Internal diameter of the tubes, Di
II. Length of the tubes, L
III. Number of tubes, n
IV. Mass flow rate of the Coolant (water), $\dot{m}_{water}$

Fixed values have been assumed for other parameters such as,

Effective Diameter of the Shell	0.144	Meters
Clearance between the tubes	0.00065	Meters
Pitch between the tubes	0.00065	Meters
Number of Baffles	3	Number

Table 5.1Assumed constants for TLBO Implementation

5.4.1 Initialization of parameters:

Rand() function of CPP Math library is used to initialize the parameters. For demonstration purpose, following are the optimization parameters considered in this case

- Population size = 10
- Number of generations = 150
- Number of design variables = 4
- Range of design variables

- $1mm < Di < 8mm$
- $50mm < L < 120mm$
- $250 < n < 350$
- $0.5\, kg/s < \dot{m}_{water} < 2\, kg/s$

In terms of TLBA language, there are four subjects and number of students (which is also the number of optimal solution sets) is considered to be 50.The initial data set obtained is given as

Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0011	0.006	291	0.70	2601	44.59	3637.10	0.00000
0.0001	0.028	253	0.80	10710	43.59	1639.79	0.00004
0.0021	0.028	268	0.50	9134	35.01	123.50	0.00027
0.0041	0.037	295	0.40	5479	34.32	1917470000.00	0.00000
0.0041	0.082	252	0.10	13566	31.98	72.16	0.00024
0.0031	0.043	266	0.90	13269	28.48	113.81	0.00034
0.0021	0.046	254	0.60	19174	27.84	480.44	0.00020
0.0041	0.092	282	0.20	10483	23.62	222.33	0.00009
0.0041	0.062	292	0.90	7239	22.62	908.68	0.00003
0.0031	0.096	271	0.70	7783	10.78	42.30	0.00014

Table 5.2 Initial Data set

For demonstration purpose, two parameters (internal diameter and Length) are plotted.

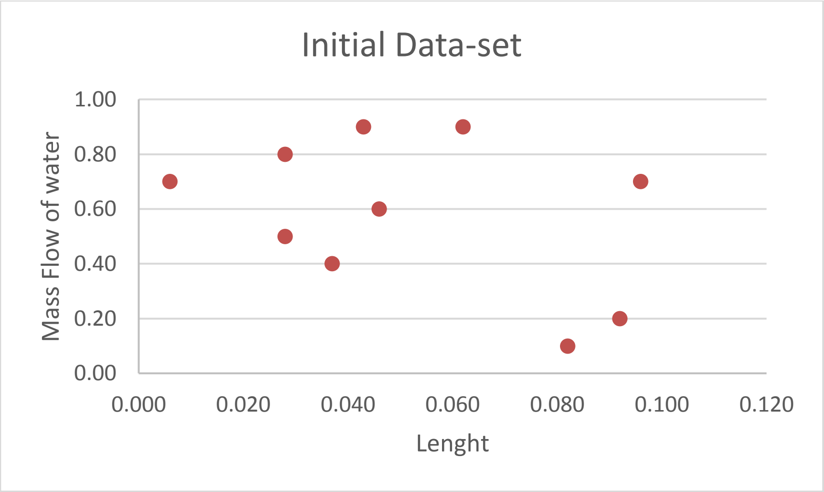

Figure 5.1Data Distribution-Initial Data Set

5.4.2 Teaching mode

5.5 Objective value for each data set i.e. each student is calculated using the

5.6 Objective Function Normalization

Since dead volume, pressure drop and heat transfer have difference units and also since there is difference in magnitudes, the objective function has to be normalized. For normalization, average pressure drop and average dead volume are used as reference.

Normalized objective function is finally given as:

Equation 5.3. The student having the best objective value is chosen as the teacher. Mean of each subject (variable) is calculated.

Difference of each parameter is calculated as follows

Equation 5.4

$$Di_{i,difference} = r\left(Di_{i,Teacher} - \left(T_f \; Di_{mean}\right)\right)$$

$$L_{i,difference} = r\left(L_{i,Teacher} - \left(T_f \; L_{mean}\right)\right)$$

$$n_{i,difference} = r\left(n_{i,Teacher} - \left(T_f \; n_{mean}\right)\right)$$

$$\dot{m}_{i,difference} = r\,(\dot{m}_{i,Teacher} - (T_f \; \dot{m}_{mean}))$$

r is any random number between 0 and 1. T_f is either 1 or 2 chosen randomly.

The difference is added to each of the student for corresponding subject (variable) as shown below.

Equation 5.5

$$Di_i = Di_i + Di_{i,difference}$$

$$L_i = L_i + L_{i,difference}$$

$$n_i = n_i + n_{i,difference}$$

$$\dot{m}_i = \dot{m}_i + \dot{m}_{i,difference}$$

5.6.1 Learning mode

In learning mode randomly any two students ($X \; and \; Y$) are selected and the following changes are followed

Equation 5.6 $if\ F[x] < F[y]$

$$Di_x = Di_x + r\,(Di_x - Di_y)$$

$$L_x = L_x + r\,(L_x - L_y)$$

$$n_x = n_x + r\,(n_x - n_y)$$

$$\dot{m}_x = \dot{m}_x + r\,(\dot{m}_x - \dot{m}_y)$$

Equation 5.7 $if\ F[x] > F[y]$

$$Di_x = Di_x + r\,(Di_y - Di_x)$$

$$L_x = L_x + r\,(L_y - L_x)$$

$$n_x = n_x + r\left(n_y - n_x\right)$$

$$\dot{m}_x = \dot{m}_x + r\,(\dot{m}_y - \dot{m}_x)$$

5.6.2 Comparison:

After each phase, each student is compared with the previous solution. The new solution is only accepted if it is better than the previous one.

Variation of design variables and objective function after generation 1, 5, 10, 15, 20, 50 and final generation are shown below:

For generation 1							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0027	0.025	289	0.54	6685	36.14	220.20	0.00004
0.0016	0.097	254	0.76	19394	11.56	11011.90	0.00005
0.0036	0.036	288	0.46	8048	33.52	80.91	0.00011
0.0041	0.049	294	0.64	10527	28.58	56.20	0.00019
0.0033	0.068	253	0.30	11351	26.54	283.91	0.00015
0.0026	0.097	258	0.82	19902	8.92	1189.74	0.00013
0.0021	0.046	254	0.60	10710	27.84	1639.79	0.00004
0.0039	0.093	279	0.32	15148	19.29	153.61	0.00031
0.0046	0.061	310	1.15	13509	22.14	38.92	0.00031
0.0023	0.136	252	0.75	24501	1.76	3459.72	0.00014

Table 5.3Data Set- Generation 1

For generation 5							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)

0.0019	0.081	265	0.72	17465	14.27	4348.04	0.00006
0.0015	0.117	254	0.83	22389	5.68	23609.40	0.00005
0.0035	0.038	286	0.47	8490	32.62	98.97	0.00010
0.0024	0.116	258	0.77	22375	4.07	2208.48	0.00013
0.0027	0.115	260	0.52	19765	9.51	1211.14	0.00017
0.0030	0.113	254	0.87	22161	4.43	756.73	0.00020
0.0020	0.114	254	0.77	21915	5.10	4657.36	0.00009
0.0036	0.136	269	0.33	19738	10.36	322.61	0.00038
0.0021	0.124	254	0.80	23503	1.89	4706.50	0.00011
0.0023	0.136	252	0.75	24501	1.76	3459.72	0.00014

Table 5.4 Data Set- Generation 5

For generation 10							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0022	0.103	259	0.76	20619	7.64	2677.74	0.00010
0.0022	0.129	254	0.77	23928	0.97	3658.39	0.00013
0.0023	0.126	253	0.82	23979	0.76	3183.89	0.00013
0.0023	0.130	254	0.75	23906	0.69	2899.85	0.00014
0.0027	0.115	260	0.52	19765	9.03	1211.14	0.00017
0.0026	0.125	253	0.81	23660	0.82	1723.83	0.00016
0.0024	0.132	254	0.74	24018	0.64	2726.06	0.00015
0.0024	0.135	254	0.72	24139	0.44	2724.03	0.00015
0.0022	0.135	252	0.76	24407	1.05	3563.58	0.00013
0.0023	0.131	253	0.77	24122	0.91	2967.48	0.00014

Table 5.5 Data Set- generation 10

For generation 15							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0023	0.129	254	0.78	23990	1.08	3069.71	0.00013
0.0023	0.130	254	0.76	24043	0.86	3139.22	0.00014
0.0023	0.126	253	0.82	23979	0.76	3183.89	0.00013
0.0023	0.130	254	0.75	23906	0.69	2899.85	0.00014
0.0026	0.142	253	0.62	23978	0.66	1969.72	0.00019
0.0026	0.128	253	0.81	23977	0.59	1712.00	0.00017
0.0024	0.132	254	0.74	24018	0.64	2726.06	0.00015
0.0024	0.135	254	0.72	24139	0.44	2724.03	0.00015
0.0023	0.128	254	0.79	24003	0.78	3097.72	0.00013
0.0029	0.128	252	0.81	23913	0.50	1066.72	0.00021

Table 5.6Data Set- Generation 15

For generation 20

Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0023	0.129	253	0.77	24010	0.80	2795.20	0.00014
0.0023	0.130	254	0.76	24043	0.86	3139.22	0.00014
0.0023	0.126	253	0.82	23979	0.76	3183.89	0.00013
0.0023	0.130	254	0.75	23906	0.69	2899.85	0.00014
0.0027	0.128	252	0.81	24019	0.44	1415.27	0.00018
0.0027	0.126	252	0.84	23930	0.53	1341.13	0.00018
0.0024	0.132	254	0.74	24018	0.64	2726.06	0.00015
0.0024	0.135	254	0.72	24139	0.44	2724.03	0.00015
0.0025	0.127	253	0.82	23988	0.62	2064.04	0.00015
0.0029	0.128	252	0.81	23913	0.50	1066.72	0.00021

Table 5.7Data Set- Generation 20

For generation 50							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0028	0.128	251	0.82	24004	0.56	1149.05	0.00020
0.0025	0.130	253	0.77	24045	0.70	2223.28	0.00016
0.0029	0.128	251	0.84	23960	0.38	949.04	0.00022
0.0027	0.128	252	0.81	24015	0.47	1330.32	0.00019
0.0027	0.128	252	0.81	24019	0.44	1415.27	0.00018
0.0027	0.127	252	0.83	24000	0.41	1304.27	0.00019
0.0024	0.132	254	0.74	24018	0.64	2726.06	0.00015
0.0024	0.135	254	0.72	24139	0.44	2724.03	0.00015
0.0028	0.128	252	0.81	23941	0.50	1233.03	0.00019
0.0029	0.128	252	0.81	23912	0.48	1061.98	0.00021

Table 5.8Data Set- Generation 50

Optimal Set							
Inner Diameter (m)	Length (m)	No.of tubes	Mass flow of Water (kg/sec)	heat transfer (W)	Objective Function	Pressure Difference (pa)	Dead Volume (m^3)
0.0029	0.128	251	0.83	23991	0.42	1053.66	0.00021
0.0029	0.128	251	0.83	24002	0.41	1086.07	0.00021
0.0029	0.128	251	0.84	23960	0.38	949.04	0.00022
0.0027	0.128	252	0.81	24015	0.47	1330.32	0.00019
0.0027	0.128	252	0.81	24019	0.44	1415.27	0.00018
0.0027	0.127	252	0.83	24000	0.41	1304.27	0.00019
0.0029	0.128	251	0.83	23998	0.44	1043.06	0.00021
0.0024	0.135	254	0.72	24139	0.44	2724.03	0.00015
0.0029	0.128	251	0.83	24000	0.44	1088.79	0.00021
0.0029	0.128	252	0.81	23912	0.48	1061.98	0.00021

Table 5.9 Data Set- Optimal Set

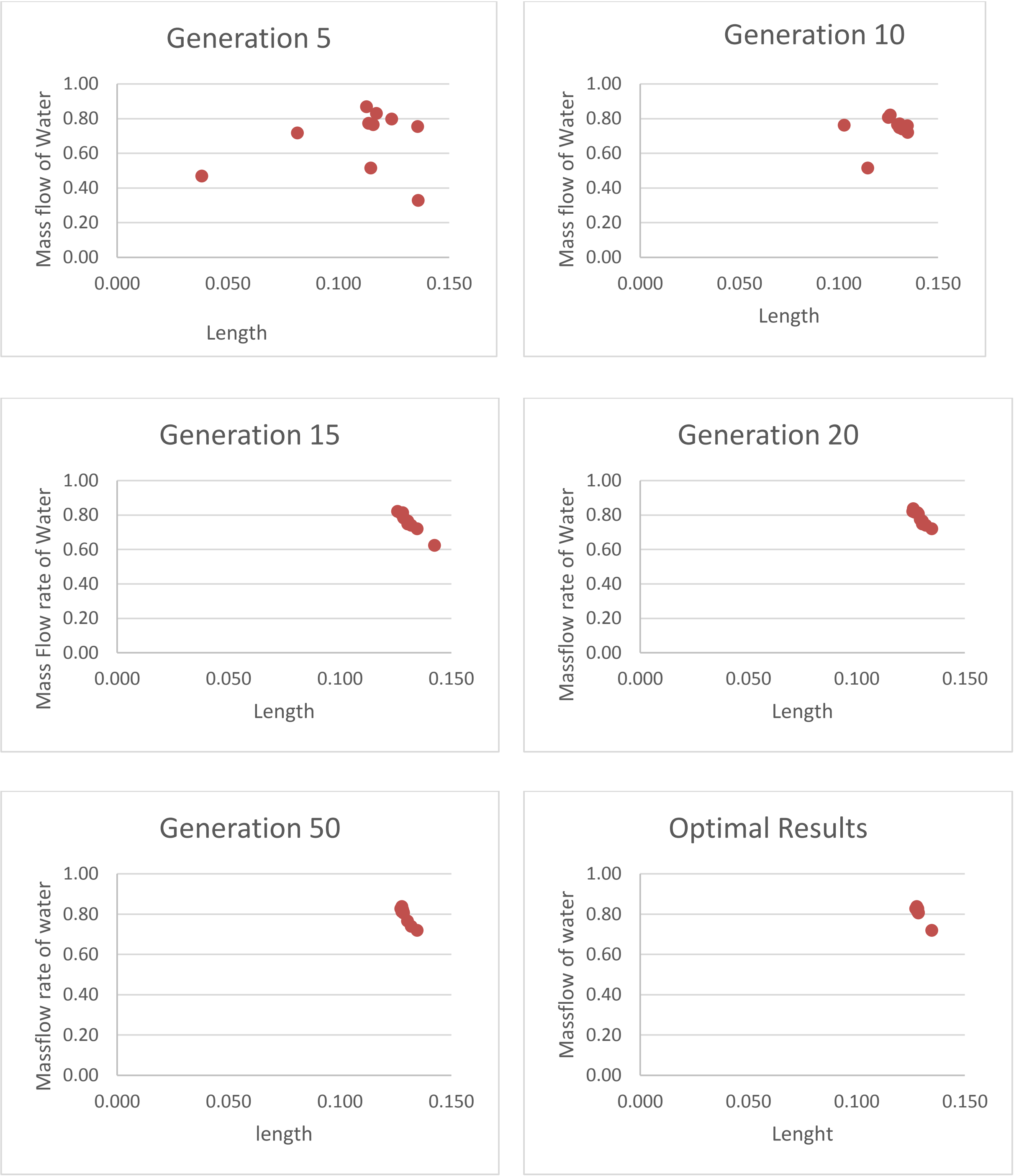

Figure 5.2 Data Distribution as solution converge

Chapter 6

6 Results and Discussion

6.1 Results

In this chapter Pareto optimal solutions obtained for various data ranges are presented. A solution is called Pareto optimal if none of the objective functions can be improved in value without degrading the others. It has been observed that with different input data ranges, different outputs are obtained. Therefore following four cases are tested and shown here with different data ranges. The average of each solution set is then compared with the existing design.

6.2 Data Sets

6.2.1 Case 1:

Range of design variables

- $0mm < Di < 5mm$
- $0mm < L < 100mm$
- $250 < n < 350$
- $0\,kg/s < \dot{m}_{water} < 1\,kg/s$

	Inner Diameter (m)	Length (m)	No. of tubes	mass flow of water (kg/sec)	heat transfer (W)	Pressure Difference (pa)	Dead Volume (m^3)
Existing Design	0.00472	0.122	384	0.59	24242.5	45.0389	0.000819
w=0.9	0.00312	0.138	260	0.62	23974.89	738.85	0.000278
w=0.5	0.0027	0.128	251	0.81	240003	1305.64	0.000197

Table 6.1 Results- Case 1

6.2.2 Case 2:

Range of design variables

- $0mm < Di < 5mm$
- $50mm < L < 100mm$
- $300 < n < 350$
- $0\,kg/s < \dot{m}_{water} < 2\,kg/s$

	Inner Diameter (m)	Length (m)	No. of tubes	mass flow of water (kg/sec)	heat transfer (W)	Pressure Difference (pa)	Dead Volume (m^3)
Existing Design	0.00472	0.122	384	0.59	24242.5	45.0389	0.000819
w=0.9	0.00541	0.123	319	1.2	23913	54.21	0.000987
w=0.5	0.0028	0.091	312	1	23810	975.28	0.000192

Table 6.2 Results Case 2

6.2.3 Case 3:

Range of design variables

- $0mm < Di < 10mm$
- $0mm < L < 100mm$
- $300 < n < 350$
- $0\,kg/s < \dot{m}_{water} < 2\,kg/s$

	Inner Diameter (m)	Length (m)	No. of tubes	mass flow of water (kg/sec)	heat transfer (W)	Pressure Difference (pa)	Dead Volume (m^3)
Existing Design	0.00472	0.122	384	0.59	24242.5	45.0389	0.000819
w=0.5	0.0024	0.091	310	1.4	23965	1051	0.000138
w=0.9	0.0051	0.111	391	1.2	23983	32.47	0.000931

Table 6.3Results Case 3

6.2.4 Case 4:

Range of design variables

- $0mm < Di < 8mm$
- $50mm < L < 100mm$
- $250 < n < 350$
- $0\,kg/s < \dot{m}_{water} < 1\,kg/s$

	Inner Diameter (m)	Length (m)	No. of tubes	mass flow of water (kg/sec)	heat transfer (W)	Pressure Difference (pa)	Dead Volume (m^3)
Existing Design	0.00472	0.122	384	0.59	24242.5	45.0389	0.000819
w=0.9	0.0032	0.119	208	1.7	23898	745.33	0.000208
w=0.5	0.0023	0.08	331	1.78	23985	1126.41	0.000116

Table 6.4 Results- Case 4

6.3 Discussion

Four cases show one common trait. Even with weighing factor as high as 0.9 (90% importance to pressure drop), the pressure drop given by the implementation of TLBO is higher than the existing design (except in one case). But, there are significant changes in dead volume. This shows that the existing design has not been designed to minimize both pressure drop and dead volume simultaneously. But, the optimal solution obtained by implementing TLBO shows promising results in terms of finding tradeoff between dead volume and pressure drop.

Different data ranges give a different set of optimal solution depending on the value of weighing factor. This phenomenon is due to the solution converging to local minima. So, the decision maker needs to make wise decisions during selected data ranges. Selection of data ranges might depend on industry standards, manufacturability, geometric limitations and other such factors.

Finding an optimal value for the coefficient c in Equation 5.3 is an important aspect while implementing an optimization algorithm involving constraints. Selecting a proper value for c is a trial and error process until the required constraint is satisfied.

Chapter 7

7 Conclusions and Future Directions

7.1 Conclusions

The development and optimization of promising technologies like VHP, increase the scope of energy savings in Heating and air-conditioning industry. The efficiency of a VHP is effected by the design of the heat exchangers which are integral to the machine. So Design optimization of heat exchangers play a critical role.

Some wrong assumptions to the values of design variables at the start of the design process may lead to waste of valuable time and resources before design parameters are enumerated for efficient heat exchanger. This work discussed the use of optimization technique which can be used at the start of the design process, thus giving a good start to the design process where the primary objectives are just not only efficient heat transfer but also other objectives. In the case of VHP heat exchangers, the other objectives are minimizing the dead volume and pressure loss of the working fluid across the heat exchanger. This tool can potentially save a significant amount of time and resources in the design process.

The developed optimization program will be useful for further design process and also in optimizing the existing design. The decision maker now has in his hand, a very useful tool which can give optimal results instantly and thus have more power while choosing a better design.

7.2 Future Direction

Optimization is never ending phenomenon. Any optimization technique and approach can be further tuned to get better results. This section describes some ways in which the above illustrated program can be made better.

Static penalty method has been used in the present approach for constraint handling. Applied mathematics world has developed lot of techniques that can handle constraints in much better way. One such method is dynamic penalty method.

Scalarization method has been used to handle multi-objectives. This method does not always guarantee a global minima/maxima. It has tendency to converge at local minima/maxima. More effective methods may be implemented to get better results.

Modified forms of TLBO can be implemented. Multiple teachers, instead of just one teacher can be considered which might help in converging much faster.

References

Berry, R. (1973). *Modular Cryogneic Refrigerators in closed Cycle Cryogenic Cooler Technology and Applications.* Dayton, Ohio: Pg: 119-136.

Bush, V. (1938). *US Patent No. 2127286.*

Bush, V. (1939). *US Patent No. 2157229.*

Carlsen, H. (1989). Development of Gas Fired Vuilleumier Heat Pump for Residential Heating. *CH2781-3/89/0000-2257 IEEE*.

Cohen, B. D. (1973). *Design and Development of a Free Displacer/Thermal Compressor Cyrogenic Refrigeration System.* Fort Belvoir, Virginia: N.V.L.U. Contrace N0: DAAK 02-71-C-0421, Editor, Philips Laboratories.

Daniels, A. d. (1971). *Miniature Refrigeration for Electronic Devices.* Philips Tech. Review 32(2)., 49-56.

Doody, R. (1971). *Two Stage Vuilleumier Cycle Cryogening Refrigerator System for Advanced Forward Looking Infrared Application.* Hughes Aircraft co., LTD.

Dynamic Programming. (n.d.). Retrieved from Wikepedia: http://en.wikipedia.org/wiki/Dynamic_programming

Holman, J. P. (2002). *Heat Transer, 9th edition.* McGraw-Hill.

Magee, F. D. (1968). *Vuilluemier Cycle Cryogening Refrigerator Development.* AFFDL-TR-68-2, Hughes Aircraft Co., LTD.

Multi-objective optimization - Wikipedia, the free encyclopedia. (n.d.). Retrieved from http://en.wikipedia.org/wiki/Multi-objective_optimization

Pitcher, G. (1970). *Design and Development of Laboratory Model Vuilluemier Cycle Cryogenic Refrigerator.* Fort Belvoir, Virginia: D.P.N.P Report No:243, Contract No. DAAKo2-69-C0364, N.V.L. USAECOM, Philips Laboratories.

Pitcher, G. (1973). *Mechanical Life of Space Coolers in Closed Cycle Cryogenic Technology and Applications.* Dayton, Ohio: AFFDL-TR-75-114, Editor, Wright-Patterson Air Force Base.

Pitcher, G. d. (1970). *Miniature Vuilleumier Cycel Refigerator. 15.P.447-451.*

R Venkata Rao, V. P. (2012). An Elitist teaching-learning based optimization algorithm for solving complex constrained optimization problems. *International Journal of Industrial Engineering Computations*.

Russo, S. (1976). *Study of Vuilleumier Cycle Cryogeninc Refrigerator for Detector Cooling in the Limb Scanning Infrared Radiometer.* Culver City, CA: N.C. AD/N77-1211, Editor, Hughes Aircraft Co.

Schlutz, S. T. (1995). Experimental investigation of a free-piston vuilleumier Refrigerator. *Butterworth-Heinemann Ltd and IIR*.

Sites, T. C.-G. (n.d.). Retrieved from sites.google.com/site/tlborao/

Thomas Baeck, D. F. (1995). *Handbook of Evolutionary Computation.* Pittsburg: A joint publication of Oxford University press and Institute of Physics Publishing.

Vuilleumier, R. (1918). *US Patent No. 1275507.*

White, R. (1973). *Program for Predicting V-M cooler off Design Performance, In Closed Cycle Cryogenic Cooler Technology Application.* Dayton, Ohio: AFFDL-TR-73-149, Wright-Patterson Air-Force Base.

White, R. (1976). *Computer Program for Optimizing Three Stage Vuilleumier Cycle Crogenic Refrigerators.* Dayton, Ohio: AFFDL-TR-76-17, Editor. Wright-Patterson Air-Force Base.

White, R. (1976). *Vuilleumier Cycle Cryogenic Refrigeration.* Dayton, Ohio: AFFDL-TR-76-17, Editor, Wright-Patterson Airforce base.

Appendix A (CPP code)

```
#include <stdio.h>
#include <math.h>
#include<stdlib.h>
#include<time.h>
#include<iostream>
#include<ctime>
#include<fstream>
#include<cmath>


using namespace std;

void main()
{
       ofstream myfile;
       myfile.open("avgref_w=0.5 case 2.csv");
       int i, j, k, l;
       int a = 10;                        //population size

       double Di[50];                     //diameter of the tube
       double L[50];               //length of the tube
       double n[50];               //number of tubes
       double t[50];               //thickness of the tube
       double m_h2o[50];           //mass flowrate of water
       double Do[50];                     //outer diameter of the tube
       double V_h2o;               //velocity of water

       double m_he = 0.59;         //mass flow rate of the Helium
       //double m_h2o = ;          //mass flow rate of water
       double cp_he = 5183.1;      //heat capacity at constant pressure of helium
       double rho_he = 15;         //density of helium
       double rho_h2o = 978;
       double mu_he = 2.357e-5;    //viscosity in Pa.S
       double mu_h2o = 0.000467;
       double k_he = 0.18921;             //thermal condutivty of helium
       double k_h2o = 0.6;                //thermal conductity of water
       double Q = 24000;                  //desired heat trasfer

       double delT_he;                    //change in temp of helium
       double delT_h2o;     //change in temp of water
       double T1_h2o = 335; //inlet temp of hwater
       double T2_h2o = 345; //inle temp of water
       double T1_he = 373; //inlet temp of helium
       double T2_he;        //outlet temp of helium
       double LMTD;                //LMTD

       double h_he;                //heat transfer coefficient of heluim
       double h_h2o;        //heat trasfer coefficeint of water
       double Re_h2o;              //reynolds number of water
       double Re_he;               //reynolds number of helium
       double Nu_he;               //nusselt number of helium
       double Nu_h2o;              //nusslet number of water
       double Ui;                         //overall heat transfer coefficent
       double delP[50];                   //pressure difference
       double k_tube = 400;        //thermal conductivity of tube
```

```
       double V_dead[50];                 //dead volume
       double vel_he;                            //velocity

       double F[50];        //Objective function
       double c1 = 8.31;// , c2 = 20000;         //constant of penalty function

       double diff[4] = { 0, 0, 0, 0 };                 //array for difference mean
       double r, Tf;               //varaibles for calculating the difference
       int  x, y, q;                      //varaible for student interation
       double A[50];// , B, C, D, E;
       double Ai, Ao, Aln, Acr;

       double Dinew[50];                  //new diameter of the tube
       double Lnew[50];                   //new length of the tube
       double nnew[50];                   //new number of tubes
       double m_h2onew[50];        //new velocity of water
       double Donew[50];                  //new outer diameter of the tube

       double Fnew[50];                   //New objective function

       double D_shell = 0.144;
       //double t_baff = 0.040;
       double clearance = 0.00065;
       double pitch = 0.00065;
       double M[4] = { 0, 0, 0, 0 };
       double Dim = 0, Lm = 0, nm = 0, Mm = 0;
       double temp1;
       int teacher;
       double w = 0.5;

       double temp;
       double heat[50], constraint[50];
       double delPavg, V_deadavg, delP_ref, V_dead_ref;

       delT_h2o = T2_h2o - T1_h2o;
       delT_he = Q / (m_he*cp_he);
       T2_he = T1_he - delT_he;
       LMTD = ((T1_he - T2_h2o) - (T2_he - T1_h2o)) / log((T1_he - T2_h2o) / (T2_he -
T1_h2o));

       //srand(time(0));
       //###########################################################################
       for (i = 0; i < a; i++)     //for Diameter
              Di[i] = (0.1 + (rand() % 5))  * 0.001;    //random number between [0,5] Any
real number

       for (i = 0; i < a; i++)     //for length
              L[i] = ((rand() % 50)+50) * 0.001;        //random integer between [5, 50]

       for (i = 0; i < a; i++)     //for number of tubes
              n[i] = (rand() % 50) + 300; //random integer between (100-600)

       for (i = 0; i < a; i++)     //for thickness of the tubes
              t[i] = 0.001;//(0.5+(rand() % 300) / 100) * 0.001; //random real number

       for (i = 0; i < a; i++)     //outer diameter
              Do[i] = Di[i] + (2 * t[i]);
```

```
       for (i = 0; i < a; i++)     //velocity of water
              m_h2o[i] =  (rand() % 20) *0.1;

       //############################################################################

       //############################################################################


       for (i = 0; i < a; i++)
       {
              Acr = (3.14*0.25)*Di[i] * Di[i];   // Area of the cros-section
              Ai = 3.14* Di[i] * L[i] * n[i];          //Inner suface area of the tube
              Ao = 3.14 * Do[i] * L[i] * n[i];   //Outer surface area of the tube
              Aln = (Ai - Ao) / log(Ai / Ao);          // Log-Mean area of the tube
              vel_he = (m_he / n[i]) / (rho_he * Acr); // Velocity of the helium
              V_h2o = m_h2o[i] / (rho_h2o * D_shell * (L[i] / 3) * (clearance / pitch));
//velocity of water

              Re_he = (rho_he * vel_he * Di[i]) / mu_he;             //Reynolds number of
the Helium
              Re_h2o = (rho_h2o * V_h2o * Do[i]) / mu_h2o;    //Reynolds number of the
Water

              Nu_he = 0.023 * pow(Re_he, 0.8)* pow(0.646, 0.33);     //Nusselt number of
the Helium
              Nu_h2o = 0.386 * pow(Re_h2o, 0.592) * pow(2.56, 0.33); //Nusselt number of
the Water

              h_he = (Nu_he * k_he) / Di[i];           //Heat Transfer coefficent of the
Helium
              h_h2o = (Nu_h2o* k_h2o) / Do[i];   //Heat Transfer coefficient of the water

              delP[i] = (0.078 * pow(Re_he, -0.25))*(L[i] / Di[i]) * (rho_he*0.5) *
pow(vel_he, 2);             // change of Pressure across the tube
              V_dead[i] = 3.14 * 0.25 * L[i] * Di[i] * Di[i] * n[i];
                                          //Dead Volume
              Ui = 1 / ((1 / h_h2o) + ((t[i] * Ao) / (k_tube*Aln)) + (Ao / (Ai*h_he)));
                                   //Over all heat transfrer coefficent
              heat[i] = Ui * Aln * LMTD;
              constraint[i] = Q - heat[i];
              A[i] = abs(constraint[i] / Q);
       }
              delPavg = 0;
              V_deadavg = 0;
              delP_ref = 0;
              V_dead_ref = 0;

       for (l = 0; l < a; l++)
       {
              delPavg = delPavg + delP[l];
              V_deadavg = V_deadavg + V_dead[l];
       }

       delP_ref = delPavg / a;
       V_dead_ref = V_deadavg / a;

       for (i = 0; i < a;i++)
       {
```

```
		F[i] = ((w*delP[i]) / delP_ref) + (((1 - w)*V_dead[i]) / V_dead_ref) + (c1*
pow(A[i], 1));				//Objective Function calculation

	}

	for (i = 0; i < a; i++)
	for (j = i + 1; j < a; j++)
	{
		if (F[i] <= F[j])
		{
			temp = F[i];
			F[i] = F[j];
			F[j] = temp;

			temp = Di[i];
			Di[i] = Di[j];
			Di[j] = temp;

			temp = L[i];
			L[i] = L[j];
			L[j] = temp;

			temp = n[i];
			n[i] = n[j];
			n[j] = temp;

			temp = m_h2o[i];
			m_h2o[i] = m_h2o[j];
			m_h2o[j] = temp;

			temp = Do[i];
			Do[i] = Do[j];
			Do[j] = temp;
		}
	}

	myfile << "Initial parameters" << endl;
	myfile << "Inner Diameter" << "," << "Thickness" << "," << "Length" << "," <<
"No.of tubes" << "," << "mass flow of h2o" << "," << "heat transfer" << "," <<
"Constraint" << "," << "obj function" << "," << "del P" << "," << "dead vol" << endl;
	for (i = 0; i < a; i++)
	{

		myfile << Di[i] << "," << t[i] << "," << L[i] << "," << n[i] << "," <<
m_h2o[i] << "," << heat[i] << "," << constraint[i] << "," << F[i] << "," << delP[i] <<
"," << V_dead[i] << endl;

	}
	myfile << endl << endl;


	for (k = 0; k <= 150; k++)
	{
		// Calculation of the Mean of each variable
		for (i = 0; i < a; i++)
		{
			Dim = Dim + Di[i];
			Lm = Lm + L[i];
```

```
                     nm = nm + n[i];
                     Mm = Mm + m_h2o[i];
              }

              M[0] = Dim / a;
              M[1] = Lm / a;
              M[2] = nm / a;
              M[3] = Mm / a;

              //Assigning Teacher
              teacher = 0;
              temp1 = F[0];
              for (i = 1; i < a; i++)
              {
                     if (F[i] > temp1)
                     {
                            temp1 = F[i];
                            teacher = i;
                     }
              }

              r = (rand() % 100);                                     //random values between [0,1]
              r = r / 100;
              Tf = (rand() % 2) + 1;                                  //Either [1,2] with equal 
probablity


              //Finding the difference
              diff[0] = r*(Di[teacher] - (Tf*M[0]));
              diff[1] = r*(L[teacher] - (Tf*M[1]));
              diff[2] = r*(n[teacher] - (Tf*M[2]));
              diff[3] = r*(m_h2o[teacher] - (Tf*M[3]));

              //Teachers influence
              for (i = 0; i < a; i++)
              {
                     Dinew[i] = Di[i] + diff[0];
                     Lnew[i] = L[i] + diff[1];
                     nnew[i] = n[i] + diff[2];
                     m_h2onew[i] = m_h2o[i] + diff[3];
                     Donew[i] = Dinew[i] + (2 * t[i]);
              }

              for (i = 0; i < a; i++)
              {
                     Acr = (3.14*0.25)*Dinew[i] * Dinew[i];                  // Area of 
the cros-section
                     Ai = 3.14* Dinew[i] * Lnew[i] * nnew[i];       //Inner suface area 
of the tube
                     Ao = 3.13 * Donew[i] * Lnew[i] * nnew[i];      //Outer surface area 
of the tube
                     Aln = (Ai - Ao) / log(Ai / Ao);                                //
Log-Mean area of the tube
                     vel_he = (m_he / nnew[i]) / (rho_he * Acr);             // Velocity 
of the helium
                     V_h2o = m_h2onew[i] / (rho_h2o * D_shell * (Lnew[i] / 3) * 
(clearance / pitch)); //velocity of water
```

```
                     Re_he = (rho_he*vel_he*Dinew[i]) / mu_he;       //Reynolds number of
the Helium
                     Re_h2o = (rho_h2o * V_h2o * Donew[i]) / mu_h2o; //Reynolds number of
the Water

                     Nu_he = 0.023 * pow(Re_he, 0.8)* pow(0.646, 0.33);     //Nusselt
number of the Helium
                     Nu_h2o = 0.386 * pow(Re_h2o, 0.592) * pow(2.56, 0.33); //Nusselt
number of the Water

                     h_he = (Nu_he * k_he) / Dinew[i];        //Heat Transfer coefficent
of the Helium
                     h_h2o = (Nu_h2o* k_h2o) / Donew[i];      //Heat Transfer coefficient
of the water

                     delP[i] = (0.078 * pow(Re_he, -0.25))*(Lnew[i] / Dinew[i]) *
(rho_he*0.5) * pow(vel_he, 2);           // change of Pressure across the tube
                     V_dead[i] = 3.14 * 0.25 * Lnew[i] * Dinew[i] * Dinew[i] * nnew[i];
                                                              //Dead Volume
                     Ui = 1 / ((1 / h_h2o) + ((t[i] * Ao) / (k_tube*Aln)) + (Ao /
(Ai*h_he)));                                    //Over all heat transfrer coefficent
                     heat[i] = Ui * Aln * LMTD;
                     constraint[i] = Q - heat[i];
                     A[i] = abs(constraint[i] / Q);
              }
              delPavg = 0;
              V_deadavg = 0;
              delP_ref = 0;
              V_dead_ref = 0;

              for (l = 0; l < a; l++)
              {
                     delPavg = delPavg + delP[l];
                     V_deadavg = V_deadavg + V_dead[l];
              }

              delP_ref = delPavg / a;
              V_dead_ref = V_deadavg / a;

              for (i = 0; i < a; i++)
              {
                     Fnew[i] = ((w*delP[i]) / delP_ref) + (((1 - w)*V_dead[i]) /
V_dead_ref) + (c1* pow(A[i], 1));                      //Objective Function calculation

                     if (Fnew[i] < F[i])
                     {
                            Di[i] = Dinew[i];
                            L[i] = Lnew[i];
                            n[i] = nnew[i];
                            m_h2o[i] = m_h2onew[i];
                            Do[i] = Di[i] + (2 * t[i]);
                            F[i] = Fnew[i];
                     }

              }

              //double Ef;
              //Student interaction
```

```
        for (j = 0; j < a; j++)
        {
            x = rand() % 10;                        //random value less than of
'a'
            y = rand() % 10;                        //random values less than
'a'
            r = (rand() % 100);
            r = r / 100;
            //Ef = 1 + r;
            if (F[x]<F[y])
            {
                Dinew[x] = Di[x] + (r*(Di[x] - Di[y]));// +(r*(Di[teacher] -
(Ef*Di[x])));
                Lnew[x] = L[x] + (r*(L[x] - L[y]));// +(r*(L[teacher] -
(Ef*L[x])));
                nnew[x] = n[x] + (r*(n[x] - n[y]));// +(r*(n[teacher] -
(Ef*n[x])));
                m_h2onew[x] = m_h2o[x] + (r*(m_h2o[x] - m_h2o[y]));//
+(r*(m_h2o[teacher] - (Ef*m_h2o[x])));
                Donew[x] = Dinew[x] + (2 * t[x]);

            }
            else
            {
                Dinew[x] = Di[x] + (r*(Di[y] - Di[x]));// +(r*(Di[teacher] -
(Ef*Di[x])));
                Lnew[x] = L[x] + (r*(L[y] - L[x]));// +(r*(L[teacher] -
(Ef*L[x])));
                nnew[x] = n[x] + (r*(n[y] - n[x]));// +(r*(n[teacher] -
(Ef*n[x])));
                m_h2onew[x] = m_h2o[x] + (r*(m_h2o[y] - m_h2o[x]));//
+(r*(m_h2o[teacher] - (Ef*m_h2o[x])));
                Donew[x] = Dinew[x] + (2 * t[x]);
            }

            for (i = 0; i < a; i++)
            {
                Acr = (3.14*0.25)*Dinew[i] * Dinew[i];          // Area of
the cros-section
                Ai = 3.14* Dinew[i] * Lnew[i] * nnew[i];  //Inner suface area
of the tube
                Ao = 3.13 * Donew[i] * Lnew[i] * nnew[i]; //Outer surface area
of the tube
                Aln = (Ai - Ao) / log(Ai / Ao);                         //
Log-Mean area of the tube
                vel_he = (m_he / nnew[i]) / (rho_he * Acr);     // Velocity
of the helium
                V_h2o = m_h2onew[i] / (rho_h2o * D_shell * (Lnew[i] / 3) *
(clearance / pitch)); //velocity of water

                Re_he = (rho_he*vel_he*Dinew[i]) / mu_he;       //Reynolds
number of the Helium
                Re_h2o = (rho_h2o * V_h2o * Donew[i]) / mu_h2o; //Reynolds
number of the Water

                Nu_he = 0.023 * pow(Re_he, 0.8)* pow(0.646, 0.33);
    //Nusselt number of the Helium
```

```
                            Nu_h2o = 0.386 * pow(Re_h2o, 0.592) * pow(2.56, 0.33);
       //Nusselt number of the Water

                            h_he = (Nu_he * k_he) / Dinew[i];        //Heat Transfer
coefficent of the Helium
                            h_h2o = (Nu_h2o* k_h2o) / Donew[i];             //Heat
Transfer coefficient of the water

                            delP[i] = (0.078 * pow(Re_he, -0.25))*(Lnew[i] / Dinew[i]) *
(rho_he*0.5) * pow(vel_he, 2);           // change of Pressure across the tube
                            V_dead[i] = 3.14 * 0.25 * Lnew[i] * Dinew[i] * Dinew[i] *
nnew[i];                                                              //Dead Volume
                            Ui = 1 / ((1 / h_h2o) + ((t[i] * Ao) / (k_tube*Aln)) + (Ao /
(Ai*h_he)));                                     //Over all heat transfrer coefficent
                            heat[i] = Ui * Aln * LMTD;
                            constraint[i] = Q - heat[i];
                            A[i] = abs(constraint[i] / Q);
                     }
                     delPavg = 0;
                     V_deadavg = 0;
                     delP_ref = 0;
                     V_dead_ref = 0;

                     for (l = 0; l < a; l++)
                     {
                            delPavg = delPavg + delP[l];
                            V_deadavg = V_deadavg + V_dead[l];
                     }

                     delP_ref = delPavg / a;
                     V_dead_ref = V_deadavg / a;

                     for (i = 0; i < a; i++)
                     {
                            Fnew[i] = ((w*delP[i]) / delP_ref) + (((1 - w)*V_dead[i]) /
V_dead_ref) + (c1* pow(A[i], 1));                       //Objective Function calculation

                            if (Fnew[i] < F[i])
                            {
                                   Di[i] = Dinew[i];
                                   L[i] = Lnew[i];
                                   n[i] = nnew[i];
                                   m_h2o[i] = m_h2onew[i];
                                   Do[i] = Di[i] + (2 * t[i]);
                                   F[i] = Fnew[i];
                            }

                     }


              }

              if (k == 1 || k == 5 || k == 10 || k == 15 || k == 20 || k == 50 || k ==
150)
              {
                     for (i = 0; i < a; i++)
                     {
                            Acr = (3.14*0.25)*Di[i] * Di[i];   // Area of the cros-section
```

```
                            Ai = 3.14* Di[i] * L[i] * n[i];          //Inner suface area
of the tube
                            Ao = 3.14 * Do[i] * L[i] * n[i];  //Outer surface area of the
tube
                            Aln = (Ai - Ao) / log(Ai / Ao);          // Log-Mean area of
the tube
                            vel_he = (m_he / n[i]) / (rho_he * Acr);  // Velocity of the
helium
                            V_h2o = m_h2o[i] / (rho_h2o * D_shell * (L[i] / 3) *
(clearance / pitch)); //velocity of water

                            Re_he = (rho_he * vel_he * Di[i]) / mu_he;
       //Reynolds number of the Helium
                            Re_h2o = (rho_h2o * V_h2o * Do[i]) / mu_h2o;    //Reynolds
number of the Water

                            Nu_he = 0.023 * pow(Re_he, 0.8)* pow(0.646, 0.33);
       //Nusselt number of the Helium
                            Nu_h2o = 0.386 * pow(Re_h2o, 0.592) * pow(2.56, 0.33);
       //Nusselt number of the Water

                            h_he = (Nu_he * k_he) / Di[i];           //Heat Transfer
coefficent of the Helium
                            h_h2o = (Nu_h2o* k_h2o) / Do[i];  //Heat Transfer coefficient
of the water

                            delP[i] = (0.078 * pow(Re_he, -0.25))*(L[i] / Di[i]) *
(rho_he*0.5) * pow(vel_he, 2);          // change of Pressure across the tube
                            V_dead[i] = 3.14 * 0.25 * L[i] * Di[i] * Di[i] * n[i];
                                                          //Dead Volume
                            Ui = 1 / ((1 / h_h2o) + ((t[i] * Ao) / (k_tube*Aln)) + (Ao /
(Ai*h_he)));                          //Over all heat transfrer coefficent
                            heat[i] = Ui * Aln * LMTD;
                            constraint[i] = Q - heat[i];

                     }
                     myfile << "For generation" << "," << k << endl;
                     myfile << "Inner Diameter" << "," << "Thickness" << "," << "Length"
<< "," << "No.of tubes" << "," << "mass flow of h2o" << "," << "heat transfer" << "," <<
"Constraint" << "," << "obj function" << "," << "del P" << "," << "dead vol" << endl;
                     for (i = 0; i < a; i++)
                     {

                            myfile << Di[i] << "," << t[i] << "," << L[i] << "," << n[i]
<< "," << m_h2o[i] << "," << heat[i] << "," << constraint[i] << "," << F[i] << "," <<
delP[i] << "," << V_dead[i] << endl;


                     }
                     myfile << endl << endl;
              }

       }



       Di[30] = 0.00472;
       Do[30] = 0.00652;
```

```
	n[30] = 384;
	L[30] = 0.122;
	m_h2o[30] = 0.59;
	t[30] = (Do[30] - Di[30]) / 2;

	Acr = (3.14*0.25)*Di[30] * Di[30]; // Area of the cros-section
	Ai = 3.14* Di[30] * L[30] * n[30];       //Inner suface area of the tube
	Ao = 3.14 * Do[30] * L[30] * n[30];      //Outer surface area of the tube
	Aln = (Ai - Ao) / log(Ai / Ao);          // Log-Mean area of the tube
	vel_he = (m_he / n[30]) / (rho_he * Acr);// Velocity of the helium
	V_h2o = m_h2o[30] / (rho_h2o * D_shell * (L[30] / 3) * (clearance / pitch));
//velocity of water

	Re_he = (rho_he * vel_he * Di[30]) / mu_he;            //Reynolds number of the
Helium
	Re_h2o = (rho_h2o * V_h2o * Do[30]) / mu_h2o;   //Reynolds number of the Water

	Nu_he = 0.023 * pow(Re_he, 0.8)* pow(0.646, 0.33);     //Nusselt number of the
Helium
	Nu_h2o = 0.386 * pow(Re_h2o, 0.592) * pow(2.56, 0.33); //Nusselt number of the
Water

	h_he = (Nu_he * k_he) / Di[30];          //Heat Transfer coefficent of the Helium
	h_h2o = (Nu_h2o* k_h2o) / Do[30];  //Heat Transfer coefficient of the water

	delP[30] = (0.078 * pow(Re_he, -0.25))*(L[30] / Di[30]) * (rho_he*0.5) *
pow(vel_he, 2);            // change of Pressure across the tube
	V_dead[30] = 3.14 * 0.25 * L[30] * Di[30] * Di[30] * n[30];
	                                         //Dead Volume
	Ui = 1 / ((1 / h_h2o) + ((t[30] * Ao) / (k_tube*Aln)) + (Ao / (Ai*h_he)));
	              //Over all heat transfrer coefficent
	heat[30] = Ui * Aln * LMTD;
	constraint[30] = Q - heat[30];



	myfile << endl << endl << "Existing design" << endl;
	myfile << "Inner Diameter" << "," << "Thickness" << "," << "Length" << "," <<
"No.of tubes" << "," << "mass flow of h2o" << "," << "heat transfer" << "," <<
"Constraint" << "," << "Obj Function" << "," << "del P" << "," << "dead vol" << endl;
	myfile << Di[30] << "," << t[30] << "," << L[30] << "," << n[30] << "," <<
m_h2o[30] << "," << heat[30] << "," << constraint[30] << "," << F[30] << "," << delP[30]
<< "," << V_dead[30] << "," << endl;
	myfile << endl;


	getchar();
}
```