1-1-2002

A new methodology for optimization of energy systems

Douglas Stinson McCorkle
Iowa State University

Recommended Citation

McCorkle, Douglas Stinson, "A new methodology for optimization of energy systems" (2002). *Retrospective Theses and Dissertations*. 20162.
https://lib.dr.iastate.edu/rtd/20162

A new methodology for optimization of energy systems

by

Douglas Stinson McCorkle

A thesis submitted to the graduate faculty
in partial fulfillment of the requirements for the degree of
MASTER OF SCIENCE

Major: Mechanical Engineering

Program of Study Committee:
Kenneth Mark Bryden (Major Professor)
Daniel Ashlock
Adrian Sannier

Iowa State University
Ames, Iowa
2002

Graduate College

Iowa State University

This is to certify that the master's thesis of

Douglas Stinson McCorkle

has met the requirements of Iowa State University

Signatures have been redacted for privacy

TABLE OF CONTENTS

LIST OF FIGURES

NOMENCLATURE

ANN	Artificial neural network
Δ	Difference between ρ and r
D	Distance metric used by a GRNN
DMP	Decision making power of an element of a neural network
e	Error for a given pattern
$E[f_i]$	Expected value for the fitness of member i
E	Batch error for entire training set
f	Fitness value
$f_X(x)$	Gaussian random variable pdf
$G(m,\sigma^2)$	Gaussian random variable distribution with mean m and variance σ^2
$g(x)$	Probability that x is greater than all other Gaussian variables
GRNN	General regression neural network
H	Heuristic used to speed up GRNN
$\bar{I}$	DMP input importance vector (after normalization)
j	Iteration count for the CFD solver
$\bar{J}$	Raw DMP input importance vector (before normalization)
k	Population member identification
m	Mating event number
N_G	Number of group members
N_I	Number of inputs
p	Pattern number in the accumulating CFD database
P	Total number of patterns
$Q(x)$	Q-function Gaussian integral approximation
ρ	Overall probability of winning
r	Probability of being selected by roulette selection

r_σ	Confidence coefficient used to validate error estimates
$\vec{T}$	Temperature profile for a given stove design
T_0	Constant used for stove fitness evaluation
v	Local weighting used by GRNN
V	Sum of v across all patterns
$\vec{x}$	Input space
y	Output, generic description
$y(\vec{x})$	GRNN estimate for $E[y \mid \vec{x}]$
$y_2(\vec{x})$	GRNN estimate for $E[y^2 \mid \vec{x}]$
$y \mid \vec{x}$	Theoretical behavior of y in the area of $\vec{x}$

ACKNOWLEDGEMENTS

I would like to thank Dr. Mark Bryden for the direction that he has given my academic and professional career and for the opportunity to work under his tutelage. I would also like to thank him for the wisdom and knowledge that he has shared. I would like to thank Dr. Daniel Ashlock for his innovative ideas that enabled this project to succeed. I would like to thank Dr. Adrian Sannier for the time he has taken to participate on my committee. I would like to thank Glen Galvin for his assistance in maintaining the computational resource need to complete this project and patience in answering questions. I would like to thank Dr. Craig Carmichael for his assistance and ideas that made this project possible and patience in working with me over the many months this project spanned. I would also like to thank my parents, Allen and Gayle, for their support and sacrifice in my educational journey. I would like to thank Nick and Kathleen for their encouragement and care of my wife during the long hours of studying and research. Finally, I would like to thank my wife Annie, for her love, support, and insight during my academic endeavors.

This work was funded in part by the U. S. Department of Energy National Energy Technology Laboratory.

ABSTRACT

This thesis presents a novel technique to significantly reduce the compute time for evolutionary optimization of systems modeled using CFD. In this scheme the typical roulette selection process is modified with a process in which competing members are represented by a Gaussian fitness distribution obtained from an artificial neural network with a feature weighted general regression neural network to create a universal approximator. This approximator develops a real-time estimate of the final fitness and error bounds during each iteration of the CFD solver. The iteration process continues until the estimated fitness and error bounds indicate that additional iterations will have a small effect on the outcome of the roulette selection process. This reduces the time required for each system call and hence reduces the overall computational time required.

CHAPTER ONE

Introduction

In many fluids systems, the response of the system to design changes cannot adequately be described with low fidelity or engineering models. Instead, high fidelity models (e.g. computational fluid dynamics (CFD)) are needed to establish the fitness of the new design. Because of this, CFD is increasingly being used to model thermal and fluid system performance as a part of the design and engineering process. However, in nearly all cases the compute time required for CFD limits its use to providing insight into limited number of specific design issues after the basic design is chosen rather than as a design and optimization tool.

Evolutionary algorithms (EAs) [1] have been applied to various optimization problems in engineering and design. In fluids and thermal systems they have been utilized to optimize a wide variety of projects including airfoils [2-4], heat exchangers [5,6], two-dimensional blade profiles [7], and missile nozzle inlets for high-speed flow [8,9]. The primary problem, in using an evolutionary algorithm to optimize a fluid and thermal system that is being modeled using CFD, is the computational cost associated with evaluating the fitness of the system. One approach that has been used is to first use low detail representation of the actual geometry to evolve the designs and then utilize a high detail model to validate and refine the solution [9,10]. In another approach, following optimization with an EA, neural networks are used with a conjugate gradient (CG) optimization routine to optimize the drag and lift of a sailing yacht fin keel. By using the information available from the EA process in a neural net to construct a global approximation of the fitness the number of CFD calls by the CG solver are minimized [11]. In another approach a neural network is used to construct a model of the search space based on the potential flow around a turbine blade. This model is then used in a EA to optimize the velocity profile on a turbine blade [12].

In this thesis a new technique for reducing the computational time needed for the evolutionary optimization of fluid systems described by high fidelity models is presented. In this scheme the typical roulette selection of co-parents is modified with a process in which competing members are represented by a Gaussian fitness distribution. This fitness is obtained from a universal approximator consisting of artificial neural network (ANN) coupled with a feature weighted general regression neural network (GRNN) [13]. This approximator develops a real-time estimate of the final fitness and error bounds while the CFD solver is running. This reduces the time required for each system call and significantly reduces the overall computational time of a fitness proportionate evolutionary system used for optimization of fluid and thermal systems based on CFD models.

CHAPTER 2

Evolutionary Optimization

In traditional EAs there are four main components that define how designs in a population are evolved. They are 1) the method of selection of two parents from the population, 2) the crossover operators, 3) the mutation operators, and 4) the method for placing the evolved children back into the general population. The selection operator controls the reproduction/survival of the population members based on their fitness values. Selection guides the search and gives direction to the evolutionary process. Popular selection methods, such as tournament selection and fitness-proportionate selection, are primarily based on distinct fitness values. In tournament selection, the best individuals from the group are selected as parents. Fitness-proportionate selection (e.g. roulette selection) is a stochastic selection method where the selection probability of an individual from the group is proportional to its fitness. The crossover and mutation operators generate two children that may be placed back in the population. The goal is to define crossover and mutation operators that enable designs to trade genetic characteristics. Several methods for placement of the children into the population are available. These include elitist replacement in which a child only replaces a member of the tournament group if they are more fit. Another approach is absolute replacement in which the children replace the least fit member of the tournament group regardless of the fitness of the children. In this thesis absolute replacement is used.

Chapter 3 describes this novel modification of the standard EA algorithm in detail for generalized use in any evolutionary optimization routine involving CFD in the fitness function and using roulette selection. In Chapter 4 the optimization problem that will be examined in this thesis is described. Chapter 5 describes the implementation of the algorithm and the results. Chapter 6 provides the conclusion and recommendations for future work.

CHAPTER 3

New Methodology

As noted above, in a typical EA coupled with a CFD solver, a fitness value f_k is evaluated for each member k of the population, and only after all the convergence criteria are met is this f_k value returned. These convergence criteria are based on resolving all the details of the flow to the maximum extent practical and are generally based on the level of the residuals and maximum change in the value of the variables from one iteration to another. This value is then treated as the absolute fitness of the system rather than the approximate fitness of the real system. A standard EA using roulette selection generates a probability

$$P(k\ roulette) = \frac{f_k}{\sum\limits_{\forall i} f_i} \qquad (1)$$

of member k being selected as a parent from a tournament group having members represented by i. Therefore, members are randomly selected for survival based on their fitness. Because of this, particular designs need only to be compared on the basis of their likeliness of selection.

When looking at a high fidelity model that utilizes an iterative process to solve a problem (e.g. CFD), it can be noted that the residual fluctuates during each iterative step of the model. If a fitness value is calculated each iteration, a different trend is seen. As the CFD solver iterates, the fitness typically settles on a solution relatively early and then slowly sharpens this value. Although the residuals may not have yet indicated convergence, the fitness value shows a predictable trend forward. This indicates that often the final fitness value is only slightly dependent on the nuances in the flow field. Given this information about the fitness as a function of iteration (Fig. 1), compute time can be saved by implementing an algorithm that predicts the final fitness value and establishes error bounds for the prediction at each iteration. When considering the accuracy of the predicted values there are two issues that need to be addressed 1) the impact on the roulette selection process and 2) impact on the absolute replacement process.

In the technique described here an ANN and a GRNN are used to predict final fitness value $E[f_k]$ and variance $VAR[f_k]$. $E[f_k]$ corresponds to the CFD solution that would have been found using the traditional convergence criteria. This speeds up the overall evolutionary process. Because the majority of the computational expense is in the CFD solver there is significant reduction in the overall computational time. In this algorithm, as the CFD solution is converging, the fitness is calculated at each iteration. For every mating event these fitness values as a function of iteration are stored in a growing database. As the database of fitness values as a function of iteration grows, a computationally inductive learning algorithm can be used to establish the predicted final fitness value $E[f_k]$ and variance $VAR[f_k]$ at each iteration of the CFD solver. With this, the probability of selection of member k, r_k, using the estimated fitness values in roulette selection is

$$r_k = \frac{E[f_k]}{\sum_{\forall i} E[f_i]} \tag{2}$$

There are various strategies that could be used to establish the stopping point for the CFD iteration process. These include:

1. Continuing the iterative process until the change in the fitness function from iteration to iteration is less than a predetermined value for a given number of iterations.

2. Continuing the iterative process until the fitness is known within a given tolerance based on the variance.
3. Continuing the iterative process until the ranking of the member within its tournament group is known within given bounds.
4. Continuing the iterative process until the probability of the member winning the roulette selection is equivalent to the probability that the member's fitness will exceed all other members fitness within its tournament group.

Method 1 is a natural choice that can be quickly implemented and has very little computational overhead. However, there is no prediction of the final fitness value. As a consequence, more CFD iterations need to be performed relative to the other three methods to achieve the same result. In contrast to Method 1, Method 2 predicts the final fitness value and the error bounds providing an opportunity for additional computational savings. However, each member is converged to the same level confidence. Provided that the error bounds are small, this provides a basis for the both roulette selection and absolute fitness replacement of the least fit member. However, additional computational savings could be gained if the stopping criteria recognized that accuracy of the predicted fitness needed varies as function of the tournament group. For example, in absolute fitness replacement the algorithm needs to accurately determine which member is to be removed from the group. If the fitness of the member currently being determined by the CFD solver lies in the middle the fitness values of its tournament group, then a larger error bound is not likely to change the outcome of the replacement process. Method 3 provides a solution to this problem by determining within some confidence interval the ranking of the member based on fitness within its tournament group. The primary disadvantage of this method is that if fitness values

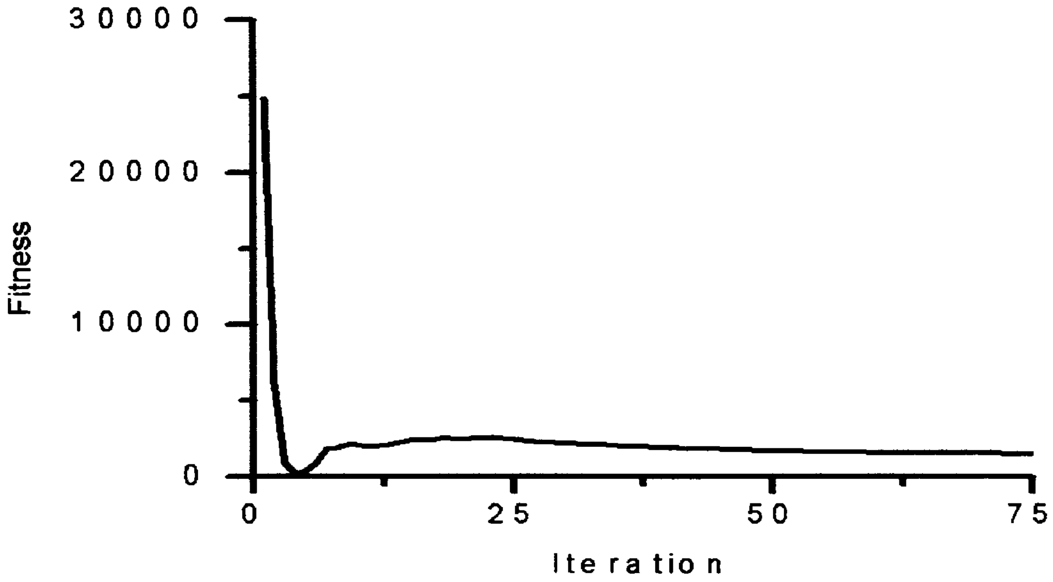


Figure 1. Fitness as a function of iteration

of two members are very close this can require substantial time to establish the needed ranking, even if the ranking does not impact the outcome of the roulette selection and absolute replacement process. Consider for example the case in which two members with fitness values near the middle of the tournament group. A very precise fitness value does not significantly change the result of the roulette selection and does not impact the absolute replacement of the least fit member. Based on this greater accuracy is needed when the fitness is high or low relative to its tournament group and less accuracy when the fitness is average relative to its tournament group.

Method 4 provides a compromise between Methods 2 and 3. In this method the iterative process is continued until the probability of winning the roulette selection process is equivalent to the probability that a member's fitness will exceed all members fitness within its tournament group, ρ_k. This can be written as $\rho_k = r_k$. When the probability that a member's fitness will exceed all members of its group is calculated at each iterative step, initially the value starts at a moderate level and then slowly asymptotically approaches 1 or 0 as the iterative process proceeds. The practical effect of this criterion is to ensure that when a member is poorly fit relative to the members of its tournament group the iterative process is continued until this ranking is clearly established. When the fitness of the member is equivalent to the average value of its tournament group fewer iterations are used. This is because the impact of the accuracy of the fitness has a small effect on the outcome. When the fitness of the member is high relative to its tournamant group more iterations are again provided because of the impact of the higher fitness on the roulette selection process and the stopping criteria. In tests utilizing Method 4 there was no significant difference in the outcome of the EA optimization process between using the traditional convergence criteria and the Method 4 convergence criteria.

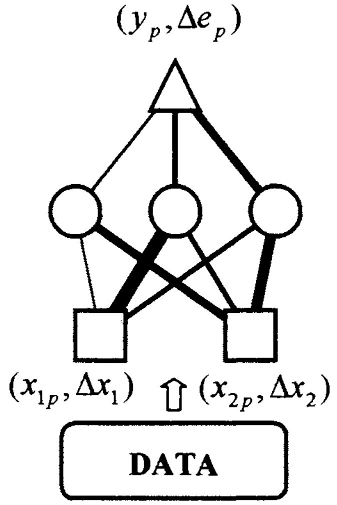
Universal Approximator

From the known fitness information up to CFD iteration j for member i, an input vector $\vec{x}_i$ can be constructed. From this, the UA can then estimate the expected value of the fitness as $m_i = E[y \mid \vec{x}_i]$, where y is a generic description of the output, or target fitness f_i. Similarly, if an estimate for $E[y^2 \mid \vec{x}_i]$ is available, then σ_i^2 can be estimated as:

$$\sigma_i^2 = \sigma_i^2[f \mid \vec{x}_i] = \left(E[f^2 \mid \vec{x}_i] - E^2[f \mid \vec{x}_i]\right) \qquad (3)$$

Which would complete the required estimate for $f_i = G(m_i, \sigma_i^2)$.

ANNs are simplified models of biological neural processes. As universal approximators, they are capable of approximating $m_i = E[f \mid \vec{x}_i]$ through a global, nonlinear mapping. They typically consist of processing units, weighted interconnections between the processing units, an activation rule to propagate signals through the network, and a learning rule to specify how the weighted interconnections are adjusted during the training phase. The most common ANN models are feed-forward networks structured in layers (Fig. 2). The networks have an input layer where the data ($\vec{x}_i$) is presented as a set of numerical patterns, one or more hidden layers to store the intermediate results of each pattern, and an output layer that contains the resulting nonlinear mapping of the network. Usually, the training phase consists of a gradient descent method, such as the scaled conjugate gradients (SCG) algorithm [14], in weight space. The weight vector $\vec{w}$ is optimized in this way to minimize the error between the desired output(s) and model output(s) over the entire training set. The



$$\frac{\delta e}{\delta x_i}(p) = \lim_{\Delta x_i \to 0} \frac{\Delta e}{\Delta x_i}(p)$$

Figure 2. Artificial neural network sensitivity analysis/structure

conjugate gradient direction is a cumulative function of steepest descent vectors from previous iterations and the current steepest descent vector $\vec{r}$. The algorithm used here combines the model-trust region approach with the conjugate gradient approach in order to give better convergence. This introduces a scalar λ that scales the Hessian matrix in an artificial way. Its main benefits are that the training regime does not contain computationally expensive line-search techniques to determine the next step size in searching for the proper network weights. The other benefit is that the algorithm requires no user-defined parameters and therefore is very easy to implement with an EA that will evolve over many hours. With SCG, the UA is able to manage the computational restraints of a growing CFD database.

Although these neural networks provide a global nonlinear mapping from the input space to the desired output target space, they do not typically provide error estimates along with their modeled output. With other methods such as decision trees and other logical representations, the output can be explained as a logical derivation from a specific set of cases that support the decision, and hence, a probability distribution can be constructed. In contrast the construction of probability distributions has been an elusive task for neural networks [15]. To construct a confidence interval on a network, a way to probe the network for these supportive cases must be established. A GRNN can establish these supportive cases. A GRNN is a feedforward neural network based on Nadaraya-Watson [16] kernel regression. The output is typically a weighted average of the target values of the training patterns close to the given input pattern, the closeness being determined by the Euclidean distance between the two patterns, where $m_i = E[y \mid \vec{x}]$ is determined by

$$E[y \mid \vec{x}] \approx y(\vec{x}) = \frac{\sum_{p=1}^{n} v_p y_p}{V} \tag{4}$$

And v_p is defined as

$$v_p = \exp\left(\frac{-D_p^2}{2\sigma_G^2}\right) \tag{5}$$

and

$$V = \sum_{p=1}^{n} v_p \tag{6}$$

D_p^2 is typically chosen to be the Euclidean distance metric between the current $\vec{x}$ and that of pattern p in the training set. Also, the single free parameter σ_G is usually determined by minimizing the predicted error sum of squares in an N-folding (training) process. In this thesis, a non-Euclidean distance metric is used so that the modified GRNN can mimic the results of the stacked ANN [17].

$$D_p^2 = \sum_{i=1}^{N_I} \left[N_I I_i (x_i - x_{p,i})\right]^2 \tag{7}$$

Where I_i represents the importance, or saliency, of input i as determined by the stacked ANN's. The method used here for determining $\bar{I}$ requires that the networks be first trained on normalized data. A sensitivity analysis is performed on the inputs across all patterns p to determine the influence J_i each input i has on the output. This is

$$J_i = \sum_{p=1}^{P} \left|\frac{\delta e_p}{\delta x_i}\right| \tag{8}$$

where e_p is the difference between the target and modeled (ANN) output. The decision making power, DMP, each input i has on the output is calculated and stored as J_i and then normalized to I_i so that the elements of the vector sum to unity:

$$I_i = \frac{J_i}{\sum_{l=1}^{N_I} J_l} \tag{9}$$

Although an ANN does not determine the supportive cases of its decision explicitly, a GRNN does this directly through the weighted averaging process. The main drawback of a GRNN is that, like kernel methods in general, it suffers from the curse of dimensionality. The advantage here is that the most useful inputs, as determined by the stacked ANN's, alter the typical Euclidean distance metric as shown above to eliminate this disadvantage. In addition, the local estimator can weigh the supportive cases so that the expected value of the fitness f and f^2 given the point $\vec{x}_i$ in design space are:

$$E[f \mid \vec{x}_i] \approx \frac{\sum_{p=1}^{P} v_p f_p}{V} \tag{10}$$

$$E[f^2 \mid \vec{x}_i] \approx \frac{\sum_{p=1}^{P} v_p f_p^2}{V} \tag{11}$$

With this, an estimate of the uncertainty of $E[f \mid \vec{x}_i]$ is obtained from Eq. 3. This completes the overall fitness estimate. The modeling techniques of both an ANN and GRNN are combined to do this. However, as P increases, the local estimator begins to dominate the overall computation time of the algorithm. To correct for this while still maintaining an accurate estimate for the converged fitness, a heuristic is used.

$$H(\vec{x}) = \sum_{i=1}^{N_I} |I_i x_i| \qquad (12)$$

Given $\vec{x}_p$, each pattern p stored in the database has a single value H_p associated with it. Therefore, if an arbitrary $\vec{x}_i$ is presented to the GRNN for recall, the nearest Q points from the sorted list $\vec{H}$ can be determined almost instantly. This reduces the search space so that all P patterns do not need to be referenced for each $\vec{x}_i$. After the points are sorted $D_P^2(\vec{x})$ is calculated and the results are sorted. The top Q_{top} patterns from this list are chosen and σ_G is increased until $V > V_{\min}$, where $V_{\min} = 3$.

During the recall process, at a given CFD iteration j, $\vec{x}_i$ is constructed from the information up to j, and the overall universal approximator $UA(\vec{x}_i)$ predicts the fitness value at each iteration ahead until the final fitness value is reached. The best projection technique found in this study starts from Q_{top} and finds the corresponding mating events M_{top} for those patterns at one CFD iteration ahead. For each CFD iteration projected ahead after that, M_{top} is held constant and refers only to those mating events already selected.

In summary, for this algorithm there are three main modifications to a traditional GRNN. First, a sensitivity analysis from the most currently trained ANN provides an input-weighting scheme for a non-Euclidean distance metric. This improves the modeling capability of a traditional GRNN as the input dimensionality grows. Second, each pattern that is added to the model data set has an associated heuristic H_p calculated every CFD run. As the CFD database grows, this heuristic vector $\vec{H}$ is updated as a sorted list. By comparing the value $H(\vec{x})$ obtained from a given input space $\vec{x}$ to only nearby patterns from this list, the search space is significantly reduced, thus speeding up the recall process. Third, the free parameter σ_G is dynamically calculated for each presentation of $\bar{x}$, thereby eliminating the typical N-folding GRNN training process. The underlying ANN is retrained on the most current CFD data on a regular and manageable basis. The modified GRNN dynamically obtains σ_G. Therefore each new pattern added to the CFD database is immediately embedded into the universal approximator's knowledge base. This is an effective implementation of incremental learning. It also significantly reduces the time required to predict the fitness distribution compared to a traditional GRNN.

Gaussian Elite Competition

As noted previously the overall structure of this algorithm is very similar to an EA using roulette selection of co-parents except that the roulette selection scheme is modified with a Gaussian elite competition scheme. This adds two new steps. First, the universal approximators are trained as the CFD database grows (Chapter 3.1). Second, the universal approximators are used with Gaussian elite competition to establish new convergence criteria for the CFD solver. Assume that member k of the population is competing in a group consisting of n other members for selection. Instead of representing each member in the group with a single fitness value, let each member be represented by a predicted fitness $E[f_i] = m_i$, and its variance, σ_i^2. The distribution of predicted values around the final value is assumed to be Gaussian. From these values the overall probability that a member k will win, ρ_k, can be determined by considering the Gaussian distribution of all competing members.

To do this, consider the probability, $g(x)$, that a value x is greater than the values of all members of group, G, where each member is described by a Gaussian probability distribution $G(m_i,\sigma_i)$. This is

$$g(x) = \prod_{i=1}^{n} P[x > G(m_i, \sigma_i^2)] \tag{13}$$

If X is a random variable representing the fitness of member k, having a Gaussian pdf with mean m_k and standard deviation σ_k, then the pdf $f_X(x)$ and the overall probability $g(x)$ are:

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{(x-m_k)^2}{2\sigma_k^2}} \qquad -\infty < x < \infty \tag{14}$$

$$g(x) = \prod_{i=1}^{n} \Phi\left(\frac{x - m_i}{\sigma_i}\right) \tag{15}$$

where Φ is the cumulative normal distribution function. This makes the overall probability, ρ_k, that k will win.

$$\begin{aligned} \rho_k &= \int_{-\infty}^{\infty} f_X(x) g(x) dx \\ &= \frac{1}{\sqrt{2\pi}\sigma_k} \int_{-\infty}^{\infty} e^{-\frac{(x-m_k)^2}{2\sigma_k^2}} \left(\prod_{i=1}^{n} \Phi\left(\frac{x - m_i}{\sigma_i}\right)\right) dx \end{aligned} \tag{16}$$

A closed form solution of this integral is not available. However, a practical and accurate approximation to $\Phi(z)$ is available through the use of the Q-function that is defined by

$$\Phi(z) = 1 - Q(z) \tag{17}$$

where

$$Q(z)=\frac{1}{\sqrt{2\pi}}\int_{z}^{\infty}e^{-t^2/2}dt \tag{18}$$

Also, $Q(0)=1/2$ and $Q(-x)=1-Q(x)$. The Q-function has a closed form approximation for $0<x<\infty$:

$$Q(z)\cong\left[\frac{1}{(1-a)z+a\sqrt{z^2+b}}\right]\frac{1}{\sqrt{2\pi}}e^{-z^2/2} \tag{19}$$

Where $a=1/\pi$ and $b=2\pi$ [18]. The fitness estimate and variance are used obtain ρ_k using Eqs. 16-19. The CFD iteration process is halted when the further accuracy of the solution has limited impact on the outcome of the EA process.

Summary of the Algorithm

Fig. 3 summarizes the overall structure of the algorithm. First, the EA is initialized and all of the population members evaluated. Each mating event consists of first randomly selecting a parent from the population. The co-parent is chosen based on modified roulette selection from a tournament group. These randomly selected tournament groups are established at the time the EA is initialized and hence are known in advance. Crossover and mutation generate two children that will replace the least fit members of the group regardless of the fitness of the children. During each mating event two fitness evaluations are performed, one for each child. For all members that will compete against k in its next tournament, the universal approximator has already returned fitness estimates for $m_i=E[f_i]$ and $\sigma_i^2=\sigma^2[f_i]$. The CFD solver is run j_{MIN} iterations and the fitness values $f_k(j)\,\forall\, j\le j_{MIN}$ stored. From here, the algorithm steps forward one iteration at a time. For each iteration the fitness estimate and variance, $(E[f_k],\sigma^2[f_k])$, are determined. The

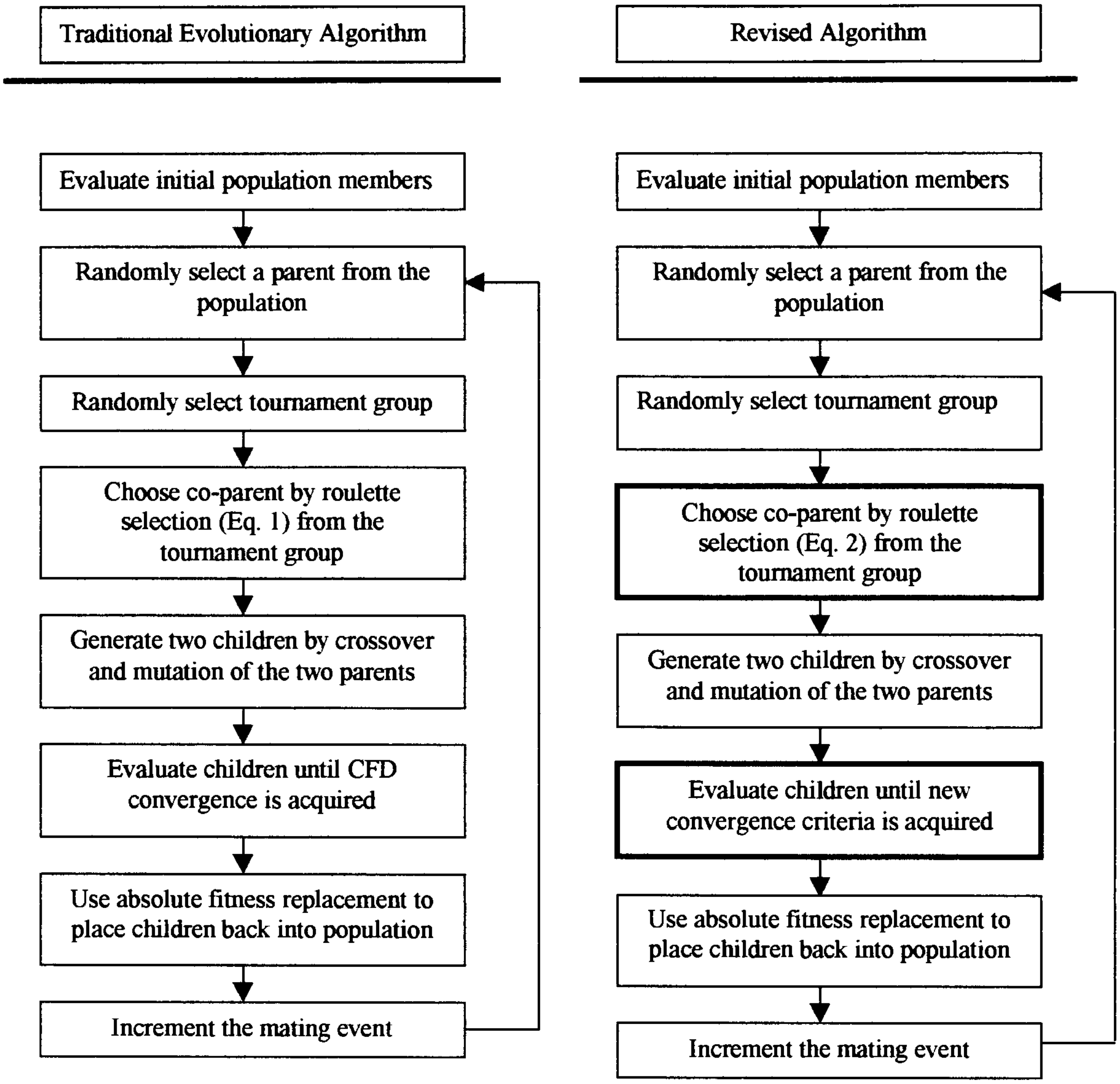

Traditional Evolutionary Algorithm
Evaluate initial population members
Randomly select a parent from the population
Randomly select tournament group
Choose co-parent by roulette selection (Eq. 1) from the tournament group
Generate two children by crossover and mutation of the two parents
Evaluate children until CFD convergence is acquired
Use absolute fitness replacement to place children back into population
Increment the mating event
Revised Algorithm
Evaluate initial population members
Randomly select a parent from the population
Randomly select tournament group
Choose co-parent by roulette selection (Eq. 2) from the tournament group
Generate two children by crossover and mutation of the two parents
Evaluate children until new convergence criteria is acquired
Use absolute fitness replacement to place children back into population
Increment the mating event

Figure 3. Evolutionary algorithm flowchart

iteration process is completed when the estimated fitness and error bounds indicate that additional iterations will have a small effect on the outcome of the EA process. The mating process continues until the EA converges.

CHAPTER 4

Energy System Model

In this thesis this algorithm is applied to the optimization of an improved plancha stove (Fig. 4). This stove is a biomass cookstove used in lower income Central American households [19, 20]. The primary purpose of this stove is to increase efficiency, reduce cooking fuel costs, and reduce the health impact of household cooking [21-24]. The current stove design has a large temperature variation on the cooking surface that limits the effectiveness of the stove for cooking. Adding baffles to the flow of the flue gas under the cooking surface can enhance the performance of the stove. However the location and size of the baffles for an effective design are unknown. A CFD model is needed for optimization because there is no simpler, lower cost solver that can predict the surface temperature of the stove. The ideal temperature distribution is a single hot spot for quickly boiling water, while maintaining the rest of the stove surface at an even temperature for cooking tortillas or simmering foods. In its current configuration without baffles the cooking surface has temperatures as cool as 150°C in some regions and as high as 600°C in others. The user sets the overall surface temperature by varying the fuel feed rate. However, the relative spatial temperature variation is generally unaffected by the fuel feed rate.

As shown in Fig. 4 this cookstove consists of an elbow shaped combustion chamber approximately 10 cm square into which small wood sticks are fed. Hot gas leaves the combustion chamber, travels under the cooking surface, and exits through a chimney. The small size of the combustion chamber relative to the size of the cooking surface is unusual and creates the large spatial variation in the temperature of the cooking surface. For this study this stove was modeled using commercial CFD software, Star-CD^{TM}. The geometry

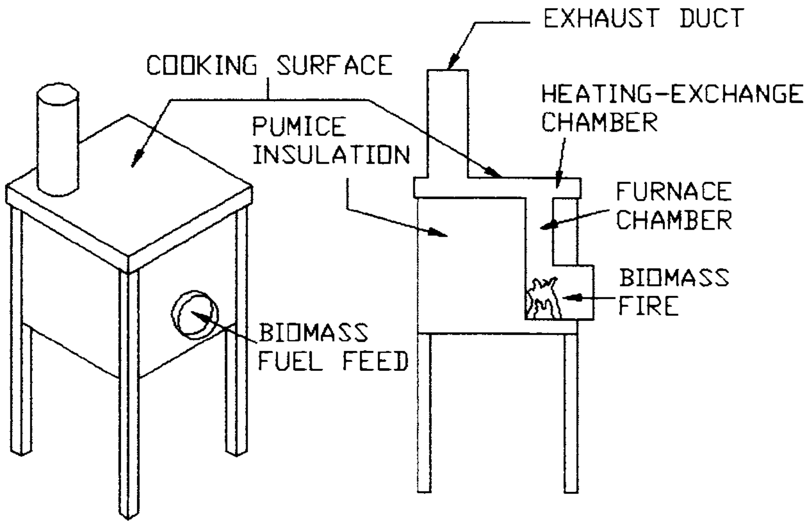


Figure 4. Plancha cookstove schematic

modeled is a simple rectangular prism representing the 54 cm × 54 cm × 2.5 cm heat exchange chamber coupled to the furnace chamber and exhaust duct (Fig. 5). Boundary conditions were determined from in-field measurements. Combustion within the combustion chamber was not modeled. Instead the boundary conditions at the inlet of the heat transfer chamber during typical cooking evolutions were measured. These were a velocity of 3.88 m/s and a temperature of 977 K. Density changes in the air due to changing the temperatures were included. Turbulence was modeled using the K-$\in$ model with an intensity of 0.1 and an entrance length of 4.8 cm. Resistance to heat transfer from the cooking surface was modeled using a heat transfer coefficient of 20 $W/m^2{\cdot}K$, a thermal conductivity of 30 $W/m{\cdot}K$, and a surface thickness of 1.6 cm. The remaining surfaces of the model were assumed to be adiabatic to simulate the pumice insulation used in the stove construction. Fig. 6a shows the CFD solution for the unbaffled EcoStove surface profile. The surface profile is in excellent agreement with the experimental data collected in Nicaragua. A full description of the computational model is given in Ref. 25. In this optimization problem, to obtain the desired temperature distribution on the cooking surface, baffles are inserted into the flow path. These baffles are perpendicular to the surface and parallel to one of the edges of the cooking surface. By diverting the flow of hot gases underneath the cooking surface the baffles change the cooking surface spatial temperature distribution (Fig. 6a).

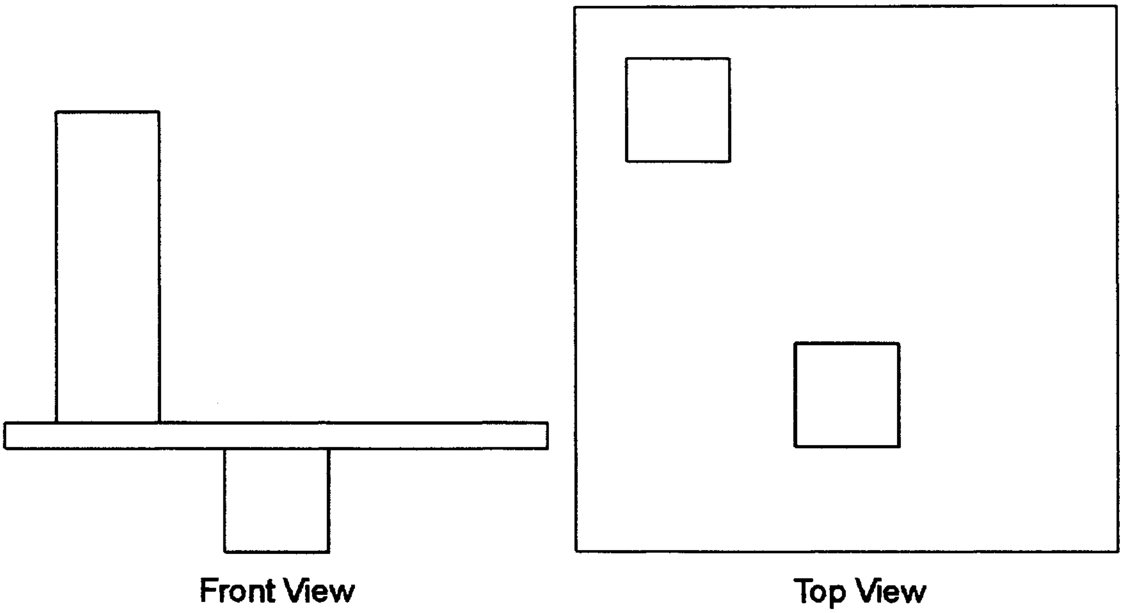


Figure 5. Computational grid for the plancha cookstove analysis

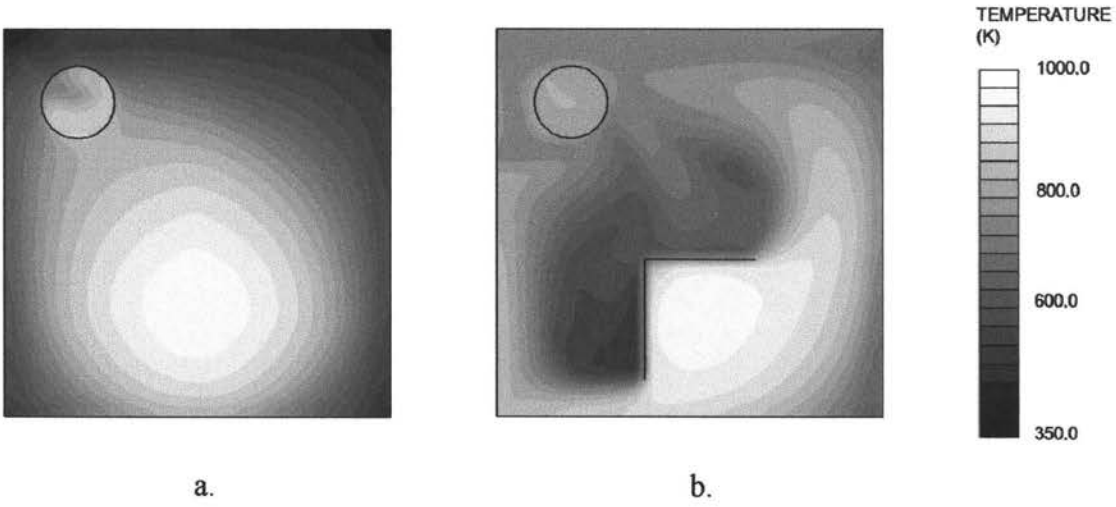


Figure 6. Stove temperature profile (a) without baffles and (b) with baffles

CHAPTER 5

Implementation and Results

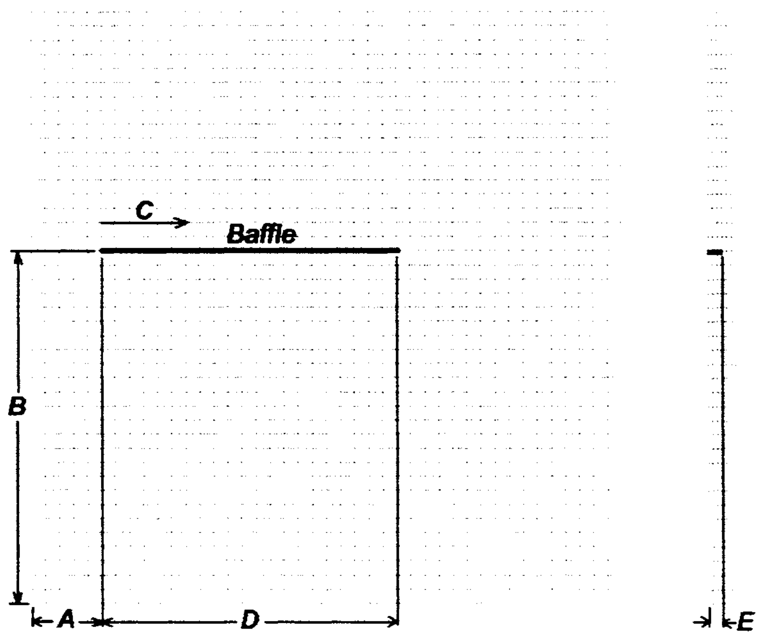
In this chapter the modified EA algorithm (Chapter 3) is used to optimize the spatial temperature distribution of the stove described in Chapter 4. To implement the modified EA algorithm there are several steps required. These steps are:

- Develop a baffle encoding structure,
- Implement the modified EA, and
- Set the parameters for configuration of the ANNs and GRNNs

To ensure that only designs that can be easily hand-built and produced are considered, baffle orientation is limited to those baffles welded perpendicular to the cooking surface and parallel to the heating chamber walls. Five values are needed to completely describe a baffle's location, length, depth, and orientation. As shown in Fig. 7, the five values are the starting x and y position of the baffle with respect to the lower left corner of the surface, the orientation of the baffle, the length of the baffle, and the depth the baffle penetrates into the flow field. Each of the values are in units of number of cells except baffle orientation, which is determined by an integer (zero or one) that represents a baffle moving right or up from the starting position. A two-dimensional array is used to represent a baffle structure with the five baffle definition values stored along a column and individual baffles stored in rows.

The EA used to optimize the plancha stove utilizes 32 creatures (baffle structures as described above) that are evolved via mutation and crossover. The fitness function for the stove problem is

$$f = \frac{\sum (T_i - T_{avg})^2}{\sum (T_i - T_{avg})^2 \Big|_{unbaffled}} \tag{20}$$



Example Baffle

A	B	C	D	E
5	25	0	21	3

Figure 7. Baffle encoding structure

where the stove surface temperature, T_i, is assessed for all surface points other than those directly above the combustion chamber entrance and below the stove chimney and T_{avg} is the average of the T_i. Based on this the fitness value of zero represents a stove with a hot spot above the inlet and a uniform temperature distribution is across the remainder of the stovetop. The fitness is normalized by the performance of an unbaffled stove. Designs with fitness values greater than 1 perform worse than an unbaffled stove. Designs with fitness values less than 1 perform better than an unbaffled stove. A fitness value of 0.2 is used as the stopping criteria for the EA. This represents a stove with approximately 90% of the stovetop temperature within 75°C of the average stovetop temperature. This is considered to be a fully effective stove because additional reductions in the fitness value do not result in additional cooking area for tortillas.

A mating event entails the following operations: selection, crossover, mutation, and replacement. First, a random creature is selected from the population. Second, a co-parent is chosen by modified roulette selection (Eq. 2). In this study the tournament group size is five creatures. Once the parents are chosen, crossover and mutation create the two children. There are two types of mutation—baffle mutation and list mutation. In baffle mutation one of the five baffle definition values is chosen randomly from a randomly chosen baffle and replaced with a new value. In list mutation a randomly chosen baffle is replaced with an entirely new random baffle. Because these stoves are handmade the number of baffles in a particular design is limited to 3 to ensure ease of manufacture. As shown in Fig. 8 crossover consists of exchanging a complete baffle definition from one parent to the other parent. The crossover

Parent 1

A	B	C	D	E
6	6	0	24	2
39	24	1	7	2
27	16	1	25	3

Parent 2

A	B	C	D	E
24	23	0	6	3
17	24	0	6	1
25	17	1	18	6

Child 1

A	B	C	D	E
6	6	0	24	2
39	24	1	7	2
25	17	1	18	6

Child 2

A	B	C	D	E
24	23	0	6	3
17	24	0	6	1
27	16	1	25	3

Figure 8. Crossover operations

point can be one of two places, either before the third baffle definition or before the second baffle definition. Once the two children are evaluated, absolute fitness replacement is used to replace the two worst members in the tournament group with the two children.

To minimize computational time required for training, the ANNs are trained on the growing database of CFD fitness data every 20 mating events. The ANN architecture consists of two hidden layers with eight nodes in the first layer and three in the second. In this case the inputs are $\lfloor f_j, f_{proj}, f_j', f_j'' \rfloor$, where f_j' and f_j'' are the first and second derivative of the fitness as a function of CFD iteration. The modified GRNN uses the closest patterns determined by $D_P^2(\bar{x})$ and incremented the value of σ_G until $V > V_{\min}$, where $V_{\min} = 3$. Fig. 9 shows the estimated values vs. the actual values for a typical EA run. R^2 for the correlation is 0.83. Fig. 10 shows the predicted error vs. the actual error for the database of patterns. An ideal model would result in a scatter plot similar to Fig. 11, where the dashed lines bound the points and the upper most line has a slope of 1 [17]. For this algorithm, $\sigma_L \approx 0$ because fitness estimates small iterations ahead result in accurate predictions with tight error bounds, and σ_H is the largest value in the respective database of patterns. As shown, 96% of the actual errors are less than or equal to the predicted 2σ confidence interval (95% confidence). Based on this the universal approximator is effective at modeling both the predicted fitness and the error bounds on the respective predictions.

To test the algorithm's compatibility with evolutionary optimization, 18 runs each with 2000 mating events were completed. On average, the algorithm was able to satisfy the modified convergence criteria in 10 CFD iterations as opposed to using the traditional convergence criteria that requires an average of 130 CFD iterations. This reduces the CFD

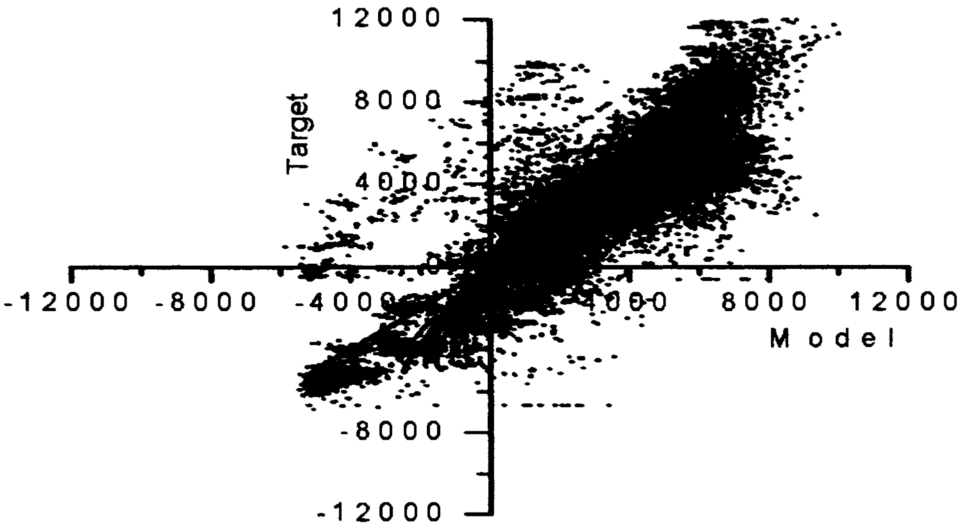


Figure 9. Target vs. model for all CFD fitness data

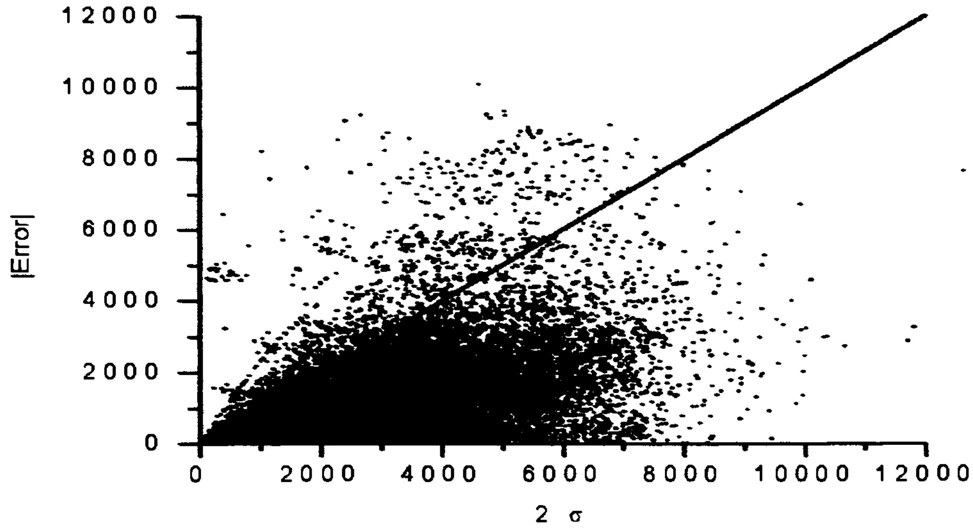


Figure 10. Validation of modeled error estimates

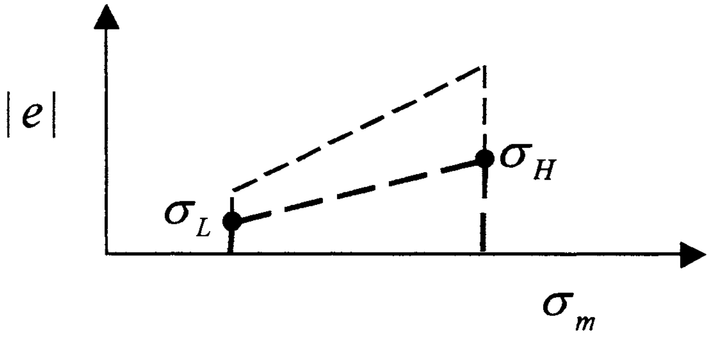


Figure 11. Best possible model for error distribution

compute time required by approximately 13 times. However, when the additional computational expense of the modified EA is included along with other overhead items an overall reduction in compute time of 8 times is realized. In a second test the both the traditional EA and the modified EA were run to convergence ($f = 0.2$) in 18 runs. From Fig. 12 it can be seen that the modified EA produced equivalent final stove design to the design produced by a traditional EA. Additionally, there was no statistical difference between the number of mating events required to reach convergence (370) between the traditional and modified EAs. The traditional EA required 2 to 2.5 days to converge. In contrast the modified EA requires approximately 12 hours to converge. This corresponds to approximately a factor of 4 decrease in computational expense. This is less than the factor of 8 mentioned earlier due to fewer than 2000 iterations needed for the convergence of the EA.

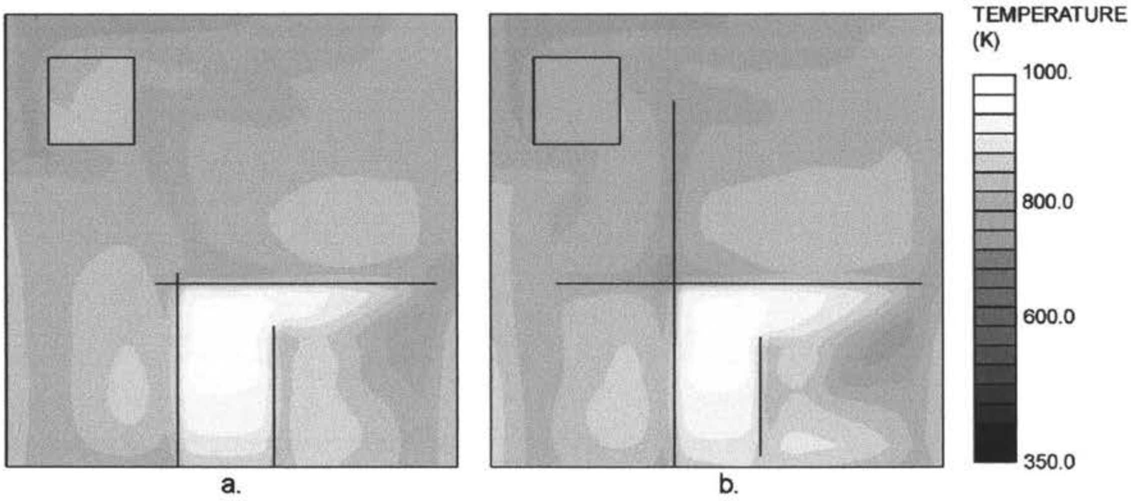


Figure 12. Comparison of temperature profiles (a) previous and (b) new algorithm

CHAPTER 6

Conclusions and Future Work

This thesis presents a novel technique to significantly reduce the compute time for evolutionary optimization of systems modeled using CFD. In this scheme the typical roulette selection process is modified with a process in which competing members are represented by a Gaussian fitness distribution. The fitness distribution is obtained from an ANN with a feature weighted GRNN to create a universal approximator. This approximator develops a real-time estimate of the fitness and error bounds while the CFD solver is running. The iteration process is completed when the estimated fitness and error bounds indicate that additional iterations will have a small effect on the outcome of the EA process. This reduces the time required for each system call and significantly reduces the overall computational time of a fitness proportionate evolutionary system used for optimization of fluid and thermal systems based on CFD models. For the thermal system model examined, in computational expense was reduced by 4 and 8 times. Future work in this area will focus on parallelization of the algorithm and the application of the algorithm to large-scale system optimization, multidisciplinary optimization, and coevolution.

REFERENCES

[1] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, Reading, Maryland, 1989.

[2] R. Mäkinen, J. Periaux, J. Toivanen, Multidisciplinary shape optimization in aerodynamics and electromagnetics using genetic algorithms, *Int. J. Numer. Methods* 30 (1999) 145-159.

[3] M. Jang, J. Lee, Genetic algorithm based design of transonic airfoils using Euler equations, in *Collection of Technical Papers–AIAA/ASME/ASCE/ASC Structures, Structural Dynamics and Materials Conference*, Atlanta, GA, 2000, 1(2), 1396-1404.

[4] D. Quagliarella, A. Vicini, Viscous single and multicomponent airfoil design with genetic algorithms, *Finite Elem. Anal. Des.* 37 (2001) 365-380.

[5] G. Fabbri, Optimization of heat transfer through finned dissipators cooled by laminar flow, *Int. J. Heat Fluid Flow* 19(1998), 644-654.

[6] T.S. Schmit, A. K. Dhingra, F.Landis, G. Kojasoy, Genetic algorithm optimization technique for compact high intensity cooler design, *J. Enhanced Heat Transfer*, 3 (1996) 281-290.

[7] M.A. Trigg, G.R. Tubby, A.G. Sheard, Automatic genetic optimization approach to two dimensional blade profile design for steam turbines, *Trans. ASME Turbomachinery*, 121 (1999), 11-17.

[8] M. Blaize, D. Knight, K. Rasheed, Automated optimal design of two-dimensional supersonic missile inlets, *J. of Propul. Power* 14 (1998), 890-898.

[9] G. Zha, D. Smith, M. Schwabacher, K. Rasheed, A. Gelsey, D. Knight, M. Haas, High performance supersonic missile inlet design using automated optimization, *J. Aircr.* 34(1997), 697-705.

[10] G. F. Foster, G. S. Dulikravich, Three-dimensional aerodynamic shape optimization and gradient search algorithms, *J. Spacecraft Rockets* 34(1997), 36-42.

[11] C. Poloni, A. Giurgevich, L. Onesti, V. Pediroda, Hybridization of a multi-objective genetic algorithm, a neural network and a classical optimizer for a complex design problem in fluid dynamics, *Comput. Methods Appl. Mech. Eng.* 186 (2000), 403-420.

[12] H. Y. Fan, Inverse design method of diffuser blades by genetic algorithms, *Proc. Inst. Mech. Eng. Part A* 212(1998), 261-268.

[13] D.F. Specht, A general regression neural network, *IEEE Transactions on Neural Networks,* 2 (1991), 568-576.

[14] M.F. Moller, A scaled conjugate gradient algorithm for fast supervised learning, *Neural Network 6* (1993), 525-533.

[15] S. Russell, P. Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, Upper Saddle River, New Jersey, 1995.

[16] E.A. Nadaraya, On estimating regression, *Theory Probab. Applic. 10*, (1964) 186-90.

[17] C.G. Carmichael, E.B. Bartlett, Stacking Diverse Models to Achieve Reliable Error Response Distributions, *International Journal of Smart Engineering System Design*, 4 (2002):31-39.

[18] P.O. Börjesson, C.E.W. Sundberg, Simple Approximations of the Error Function $Q(x)$ for Communications Applications, *IEEE Trans. Commun.* (1979) 639-643.

[19] Proleña–Nicaragua, *Alternativas Viables Para Solucionar el Problema de Demanda de Leña en la Región Las Segovias*, ed. S.F. Alves-Milho, Managua, Nicaragua, 2000.

[20] Instituto Nicaragüense de Energia (INE), Memoria INE 1997, Managua, Nicaragua, 1997.

[21] C.J. Hong, *Health Aspects of Domestic Use of Biomass Fuels and Coal in China,* Shanghai Medical University, Shanghai, China, 1994.

[22] D. Barnes, K. Openshaw, K.R. Smith, R. van der Plas, What makes people cook with improved biomass stoves?, *World Bank Technical Paper: Energy Series* 242 (1994) 1-39.

[23] P. Malhotra, Environmental implications of the energy ladder in rural India, *Boiling Point* 42 (1999) 3-5.

[24] M. Pandey, Health risk caused by domestic smoke, *Boiling Point* 40 (1998) 6-8.

[25] G. Urban, The EcoStove: A case study in thermal system optimization based on differential analysis, Masters thesis, Iowa State University, 2001.