Rice University

A Scattered Data Approximation Tool to Map Carbon Nanotube Dispersion to the Processing Parameters in Polymer Nanocomposites

by

Jonathan W. Lee

A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Master of Science

Approved, Thesis Committee:

[illegible]

Enrique V. Barrera, Committee Chair
Professor of Mechanical Engineering &
Materials Science

[illegible]

Andrew J. Meade, Jr., Thesis Director
Professor of Mechanical Engineering &
Materials Science

[illegible]

Brent C. Houchens
Assistant Professor of Mechanical
Engineering & Materials Science

Houston, Texas

December, 2008

Abstract

A Scattered Data Approximation Tool to Map Carbon Nanotube Dispersion to the Processing Parameters in Polymer Nanocomposites

by

Jonathan W. Lee

The relationship of nanocomposite dispersion was studied with the variation of dispersion techniques and other processing parameters. Examining all permutations of the various factors in the laboratory is a challenging task. In this thesis, we propose to map a correlation between inputs and output via a self-adaptive scattered data approximation method. The proposed greedy algorithm, Sequential Function Approximation (SFA), reveals the multidimensional behavior of the system, provides the sensitivity of each input, and presents the combination of inputs that is most suitable for a specific output. In this research, we have collected data from various research institutions and applied it to SFA. The results show that CNT weight percent, sonication time, CNT modification, and high shear mixing time are key factors that affect the dispersion. This thesis discusses SFA, the data, and the results in detail. This work serves as a proof of concept and recommended future work is discussed.

Acknowledgements

I would like to thank my advisors, Dr. Enrique V. Barrera and Dr. Andrew J. Meade, for their encouragement and guidance throughout this project. I also would like to thank my group members, past and present, for their friendship and support. Most notably, I'd like to thank Ankur Srivastava for helping me understand SFA and Laura Peña-Parás for the countless hours she devoted to helping me extract data and comprehend the "materials" side of things. I also want to thank Dr. Brent C. Houchens for serving on my thesis committee. Dr. Houchens has been a great professor, mentor, and friend since I enrolled at Rice.

This research was funded by the URETI Grant: NCC-1-02038 and the Welch Foundation Grant: C-1494.

Personally, I would like to thank my friends and family who have been so uplifting throughout this project. I am endlessly grateful that I have so many loved ones to stand by me through the ups and downs.

Contents

List of Tables

List of Figures

Nomenclature

b_n	=	n^{th} bias term of the approximation
c_n	=	n^{th} linear coefficient in the approximation
d	=	input dimension
D	=	dictionary of basis functions
$\vec{f}$	=	arbitrary function
G	=	greedy operator
H	=	Hilbert space
j^*	=	component index of $\vec{R}_{(n-1)}$ with the maximum absolute magnitude
n	=	number of bases
$\mathbb{R}$	=	real coordinate space
$r_n\left(\vec{\xi}\right)$	=	n^{th} stage of the function residual
$\vec{R}_n$	=	residual vector at the n^{th} stage
s	=	number of samples
$u\left(\vec{\xi}\right)$	=	target function evaluated at $\vec{\xi}$
$u_n^a\left(\vec{\xi}\right)$	=	n^{th} stage of the target function approximation evaluated at $\vec{\xi}$
$\vec{v}$	=	arbitrary function
$\left\langle \vec{f} \right\rangle_D$	=	$\sum_{i=1}^{s} f_i$ = discrete inner product of $\vec{f}$ and $\vec{1}$
$\left\langle \vec{f}, \vec{v} \right\rangle_D$	=	$\sum_{i=1}^{s} f_i v_i$ = discrete inner product of $\vec{f}$ and $\vec{v}$
$\left\lvert \vec{f} \right\rvert$	=	absolute value of $\vec{f}$

Greek Symbols

ϕ	=	basis function
$\vec{\Phi}_n$	=	vector of the n^{th} basis function
θ_n	=	angle between $\vec{\Phi}_n$ and $\vec{R}_{n-1}$

σ_n	=	n^{th} width parameter of Gaussian basis function
τ	=	user specified tolerance
$\vec{\xi}$	=	d -dimensional input of the target function
$\vec{\xi}_i$	=	i^{th} sample input of the target function
$\vec{\xi}_n^*$	=	sample input with component index j^* at the n^{th} stage

Subscripts

i	=	dummy index
j	=	index of sample number
j^*	=	component index of $\vec{R}_{(n-1)}$ with the maximum magnitude
n	=	associated with the number of bases
s	=	associated with the number of samples

Superscripts

d	=	associated with the input dimension

Chapter 1

Introduction

Composite materials (or composites for short) are materials comprised of two or more constituent materials. Typically the constituent materials have vastly different physical or chemical properties, and combine with a synergistic effect to create a new, better material. The two types of constituent materials are known as the matrix and reinforcement. At least one of each type is required. The matrix is used to hold the reinforcement in place, while the reinforcement is used to enhance the properties of the matrix.

Carbon nanotubes (CNTs) are a relatively new type of molecule comprised of only carbon atoms. The CNTs take the form of cylindrical tubes, as if sheets of graphene were rolled onto themselves. Due to their unique shape, aspect ratio, and composition, CNTs display very unique properties [1-3]. For this reason, CNTs are being employed in composites as the reinforcement constituent. CNT composites may also be referred to as "nanocomposites" because of their "nano-constituents."

Polymers are typically used in these nanocomposites as the matrix material. However, a great deal of research has been directed towards dispersing and dissolving CNTs in the matrix such that the composite is a stable mixture [4-15]. Many different approaches to polymer-CNT composite processing have been attempted, with varied success. These many approaches suggest that there are many factors that could potentially influence the ultimate product. The goal in processing polymer-CNT composites is to achieve a significant enhancement of the matrix's material properties. In essence, each experiment performed is aimed at finding the correlation between some processing parameters and the material property.

The goal is to determine how to make better composites. Rather than approaching this problem one experiment at a time, an alternative is to build a model based on existing data. High-dimensional (i.e., systems with large numbers of input variables) mapping is a tool commonly used in science and engineering. A database that contains a fixed set of input variables and output variable(s) is acquired and applied to the mapping tool. An input-output mapping is then generated based on a training set (a portion of the whole database). With the mapping, the correlations between inputs and output(s) are now expressed as a function. The function has predictive capabilities, and can be used for design purposes. This type of modeling is called function approximation or scattered data approximation. Specifically, we employ Sequential Function Approximation (SFA) in this work [16].

Due to the complex nature of composite processing, physical knowledge of the domain is limited and theoretical modeling is difficult. Polymer-CNT process design can be modeled using SFA because it only utilizes empirical data to build the model. The many processing parameters and other influencing factors are treated as inputs to the system. The output of the system can be any discernible assessment of the final product.

A mathematical model of the system would provide sensitivity information about the many processing factors. It would also save time and money by avoiding a permutation approach in which engineers would have to test a vast number of experiments. The model would also allow scientists and engineers to explore other options and processing parameters that have been overlooked.

In this work, a kernel based learning algorithm is applied to polymer-CNT composite processing as described above. A model was developed with very sparse data to achieve better than 60% accuracy. A sensitivity analysis was also performed which provided the most and least sensitive inputs. The results from the analyses were corroborated by current knowledge of the domain as well as knowledge of the algorithm.

In Chapter 2, SFA is described in detail and examined with three test cases. Chapter 3 discusses the method of applying the algorithm to polymer-CNT processing. Chapter 4 presents the results of the model and compares them against existing knowledge. Conclusion and future work are discussed in Chapter 5.

Chapter 2

Function Approximation Theory

2.1 Introduction

Let us consider a deterministic system which takes as input the vector $\vec{x} \in \mathbb{R}^d$ and returns a scalar output $y \in \mathbb{R}$. With s repeated experiments, we have a set of s input vectors, $\vec{X} = \left\{\vec{x}^{(1)}, \vec{x}^{(2)}, \ldots, \vec{x}^{(s)}\right\} \in \mathbb{R}^{d \times s}$, and the set of corresponding output values $\vec{y} = \left\{y^{(1)}, y^{(2)}, \ldots, y^{(s)}\right\} \in \mathbb{R}^s$. A portion of the data is allocated to training. The training data is often obtained by a design of experiments technique. Given the training data, we are concerned with constructing an approximation to the function which maps the input data $\vec{x}$ to the output y. The approximation problem is essentially the prediction of the output $y(\vec{x})$ given a new input vector $\underline{x}$. The output data can be categorical (binary $[-1,+1]$ or integers) yielding a classification problem or continuous thereby yielding a regression problem. The outputs we will concern ourselves with in this text will be categorical. Such a problem is in general referred to as black-box modeling. Black-box modeling generally consists of the following steps: (1) data generation, (2) model-structure selection, (3) parameter estimation, and (4) model validation [17].

Data generation, for our purposes, is more applicably termed data collection. For problems where the training data can be computationally generated, the location of the training points can be selectively placed so that the predictive capability can be optimized. In our case, the location of our training points is largely dictated by the data that we are able to collect. Selection of the training data from the collected data can be selectively chosen by various means [17]. The means by which we select our training data is random selection. Taking the mean result from several random permutations is representative of the model's performance. One selective method which we will explore is called Latin Hypercube sampling. This method will be addressed in Section 4.6.

The next step in black-box modeling involves specifying the structure of the model. In practice, this is essentially a question of whether the data is better modeled with a parametric or non-parametric model. A model is parametric if the training data is no longer needed after the model parameters are estimated. Conversely, a non-parametric model requires the training data even after the model parameters have been established [17]. The algorithm that we employ in this thesis deals with non-parametric models.

For any given model, there must be a set of unknown model parameters which must be determined. Typically the parameters are chosen in a way which will minimize error, though there are several techniques for estimating the parameters. Particularly, the choice of model type will affect the way the parameters are determined [17]. In our algorithm, the parameters are determined to minimize the residual.

Finally, we have model assessment. In model assessment, we run a diagnostic procedure to ensure the suitability of the model. In statistics, goodness-of-fit tests and diagnostic plots can be employed to check the adequacy of the model [17]. For our method, we compare our model against the remaining data, which we have reserved as testing data. The testing points which are incorrectly predicted are accumulated for an assessment of the error.

In the remainder of this text, the vector of output classes will be referred to as the target function. Generally, for a classification problem the target function is complicated and needs to be expressed in terms of easily computable functions called bases. This is a very basic problem of constructive approximation theory. Resolution of the target function may be increased by choosing more complicated basis functions at the cost of simplicity of the approximation. If the basis functions come from a linear space it is termed a linear approximation and if they are chosen from non-linear manifolds it is called non-linear approximation. The early methods employed in linear approximation required the basis to be chosen from finite dimensional linear spaces like algebraic and trigonometric polynomials. Splines, piecewise polynomials, and wavelet theory were other important contributions to this field [18]. It was soon realized that choosing the approximants from non-linear manifolds yielded better rates of approximation. Nonlinear approximation techniques, where the choice of the basis depended on the target function, were soon developed. An extensive discussion on nonlinear approximation theory is presented in reference [19]. In 1907 Schmidt [20] proposed the idea of constructing an approximation of the target function in a nonlinear fashion simultaneously keeping control of the number of terms used in the linear sum. This gave rise to *n-term approximation.* It has been studied thoroughly in the past and its supremacy over linear approximation has been established [19]. This led to the development of a nonlinear approximation technique where the target function was not only used to choose the best basis function from a collection of basis functions but was also used to determine the coefficients required for the linear sum. Doing this ensures that the approximation is the best n-term approximation possible for the given target function. Such a double stage approximation technique is often referred to as a *highly nonlinear* technique. Some of the popular algorithms belonging to this category are greedy algorithms and adaptive pursuit.

A brief discussion on greedy algorithms is presented in this dissertation because SFA falls under this category of algorithms. Assume that the target function f belongs to the Hilbert space H, and a collection of basis functions often called a dictionary D is given. For a classification problem, the training points mapped to a higher dimensional space via a kernel function act as a dictionary. Then the greedy algorithm can be expressed mathematically as follows [19,21]:

Let $u \in H$ and $\phi = \phi(u) \in D$ denote an element chosen from D such that

$$\langle u, \phi \rangle = \sup_{\phi \in D} \langle u, \phi \rangle. \tag{2.1}$$

Define the following:

$$\begin{aligned} G(u) &= \langle u, \phi \rangle \phi \\ R(u) &= u - G(u). \end{aligned} \tag{2.2}$$

Initially set $R_0(u) = u$ and $G_0(u) = 0$. Then for each $n \geq 1$:

$$\begin{aligned} G_n(u) &= G_{n-1}(u) + G(R_{n-1}(u)) \\ R_n(u) &= u - G_n(u) = R(R_{n-1}(u)). \end{aligned} \tag{2.3}$$

Here g is the basis function, G is a greedy operator, R is the residual vector, and $\langle u, \phi \rangle$ is the inner product of u with respect to ϕ. Given the dictionary is orthogonal, a basis that maximizes the absolute inner product with the target function is chosen. Its contribution is subtracted to obtain the residual and iterated.

The above-mentioned algorithm is called a Pure Greedy Algorithm [19,21]. In this work, we are concerned with Pure Greedy Algorithms only because of their simplicity and effectiveness. Work with other variations of greedy algorithms deserves a separate study.

2.2 Sequential Function Approximation

We start our approximation of the unknown target function u by noting that a continuous d-dimensional function can be arbitrarily well approximated by a linear combination of radial basis functions ϕ. Sequential Function Approximation (SFA) approximates u as

$$u_n^a\left(\vec{\xi}\right)=\sum_{i=1}^{n}c_i\left(\phi\left(\vec{\xi},\vec{\xi}_i^*,\sigma_i\right)+b_i\right) \tag{2.4}$$

with

$$\phi\left(\vec{\xi},\vec{\xi}_i^*,\sigma_i\right)=\exp\left[\ln\left[\sigma_i\right]\left(\vec{\xi}-\vec{\xi}_i^*\right)\cdot\left(\vec{\xi}-\vec{\xi}_i^*\right)\right], \tag{2.5}$$

for $0<\sigma_i<1$ where $\vec{\xi}\in\mathbb{R}^d$, $\vec{\xi}_i^*\in\mathbb{R}^d$, $\sigma_i\in\mathbb{R}$, $b_i\in\mathbb{R}$, and $c_i\in\mathbb{R}$ represent the function argument, the i^{th} basis function center, independent variables, function parameters, and linear coefficients, respectively.

The basic principles of our greedy algorithm are motivated by the similarities between the Jones [22,23] and Barron [24] iterative optimization procedures and the Method of Weighted Residuals (MWR), specifically the Galerkin method (Fletcher [25]). We can write the function residual r at the n^{th} stage of approximation as

$$\begin{aligned} r_n\left(\vec{\xi}\right) &= u\left(\vec{\xi}\right) - u_n^a\left(\vec{\xi}\right) \\ &= u\left(\vec{\xi}\right) - u_{n-1}^a\left(\vec{\xi}\right) - c_n\left(\phi\left(\vec{\xi},\vec{\xi}_n^*,\sigma_n\right) + b_n\right) \\ &= r_{n-1}\left(\vec{\xi}\right) - c_n\left(\phi\left(\vec{\xi},\vec{\xi}_n^*,\sigma_n\right) + b_n\right). \end{aligned} \tag{2.6}$$

Because we are given a finite number of samples (s) of the observed data, we can write

$$\begin{aligned} \vec{R}_{n-1} &\equiv \left\{ r_{n-1}\left(\vec{\xi}_1\right), r_{n-1}\left(\vec{\xi}_2\right), \ldots, r_{n-1}\left(\vec{\xi}_s\right) \right\} \\ \vec{\Phi}_n &\equiv \left\{ \phi\left(\vec{\xi}_1,\vec{\xi}_n^*,\sigma_n\right) + b_n, \phi\left(\vec{\xi}_2,\vec{\xi}_n^*,\sigma_n\right) + b_n, \ldots, \phi\left(\vec{\xi}_s,\vec{\xi}_n^*,\sigma_n\right) + b_n \right\}. \end{aligned} \tag{2.7}$$

Utilizing the Petrov-Galerkin approach, we select a coefficient c_n that will force the function residual to be orthogonal to the basis function and b_n using the discrete inner product

$$\begin{aligned} \vec{R}_n \cdot \vec{\Phi}_n &\equiv \left\langle \vec{R}_n, \vec{\Phi}_n \right\rangle_D \\ &= -\left\langle \vec{R}_n, \frac{\partial \vec{R}_n}{\partial c_n} \right\rangle_D \\ &= -\frac{1}{2}\frac{\partial \left\langle \vec{R}_n, \vec{R}_n \right\rangle_D}{\partial c_n} = 0, \end{aligned} \tag{2.8}$$

which is equivalent to selecting a value of c_n that will minimize $\left\langle \vec{R}_n, \vec{R}_n \right\rangle_D$. It can be proven that

$$\begin{aligned} c_n &= \frac{\left(s\left\langle \vec{\Phi}_n, \vec{R}_{n-1} \right\rangle_D - \left\langle \vec{R}_{n-1} \right\rangle_D \left\langle \vec{\Phi}_n \right\rangle_D \right)}{\left(s\left\langle \vec{\Phi}_n, \vec{\Phi}_n \right\rangle_D - \left\langle \vec{\Phi}_n \right\rangle_D \left\langle \vec{\Phi}_n \right\rangle_D \right)} \\ b_n &= \frac{\left(\left\langle \vec{R}_{n-1} \right\rangle_D \left\langle \vec{\Phi}_n, \vec{\Phi}_n \right\rangle_D - \left\langle \vec{\Phi}_n \right\rangle_D \left\langle \vec{\Phi}_n, \vec{R}_{n-1} \right\rangle_D \right)}{\left(s\left\langle \vec{\Phi}_n, \vec{R}_{n-1} \right\rangle_D - \left\langle \vec{R}_{n-1} \right\rangle_D \left\langle \vec{\Phi}_n \right\rangle_D \right)}. \end{aligned} \tag{2.9}$$

will minimize $\left\langle \vec{R}_n, \vec{R}_n \right\rangle_D$, where $\left\langle \vec{f} \right\rangle_D \equiv \sum_{i=1}^{s} f_i$. The discrete inner product $\left\langle \vec{R}_n, \vec{R}_n \right\rangle_D$, which is equivalent to the square of the discrete L_2 norm, can be rewritten with the substitution of Eq. (2.9) as

$$\left\langle \vec{R}_n, \vec{R}_n \right\rangle_D = \left\langle \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s}, \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s} \right\rangle_D \left(1 - \frac{\left\langle \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s}, \vec{\Phi}_n - \frac{\left\langle \vec{\Phi}_n \right\rangle_D}{s} \right\rangle_D^2}{\left\langle \vec{\Phi}_n - \frac{\left\langle \vec{\Phi}_n \right\rangle_D}{s}, \vec{\Phi}_n - \frac{\left\langle \vec{\Phi}_n \right\rangle_D}{s} \right\rangle_D \left\langle \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s}, \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s} \right\rangle_D} \right). \tag{2.10}$$

Recalling the definition of the cosine using arbitrary functions $\vec{f}$ and $\vec{v}$ and the discrete inner product,

$$\cos(\theta_n) = \frac{\left\langle \vec{f}, \vec{v} \right\rangle_D}{\left\langle \vec{f}, \vec{f} \right\rangle_D^{1/2} \left\langle \vec{v}, \vec{v} \right\rangle_D^{1/2}}, \tag{2.11}$$

then Eq. (2.10) can be written as

$$\begin{aligned} \left\| \vec{R}_n \right\|_{2,D}^2 &= \left\langle \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s}, \vec{R}_{n-1} - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s} \right\rangle_D \sin^2(\theta_n) \\ &= \left(\left\| \vec{R}_{n-1} \right\|_{2,D}^2 - \frac{\left\langle \vec{R}_{n-1} \right\rangle_D^2}{s} \right) \sin^2(\theta_n), \end{aligned} \tag{2.12}$$

where θ_n is the angle between $\vec{\Phi}_n$ and $\vec{R}_{n-1}$ since $\frac{\left\langle \vec{R}_{n-1} \right\rangle_D}{s}$ and $\frac{\left\langle \vec{\Phi}_n \right\rangle_D}{s}$ are scalars. With Eq. (2.12), we note that $\left\langle \vec{R}_n, \vec{R}_n \right\rangle_D \le \left\langle \vec{R}_{n-1}, \vec{R}_{n-1} \right\rangle_D$ as long as $\theta_n \ne \pi/2$, which is a very robust condition for convergence. By inspection, the minimum of Eq. (2.12) is $\theta_n = 0$, or

$$c_n\left(\phi\left(\vec{\xi}_i,\vec{\xi}_n^*,\sigma_n\right)+b_n\right)=r_{n-1}\left(\vec{\xi}_i\right) \quad \text{for } i=1,\ldots,s. \tag{2.13}$$

Therefore, to force $\left\langle \vec{R}_n,\vec{R}_n\right\rangle_D \to 0$ with as few stages of n as possible, a low dimensional function approximation problem must be solved at each stage. This involves a constrained nonlinear minimization of Eq. (2.10) in the determination of the two variables σ_n $\left(0<\sigma_n<1\right)$ and index $j^*\left(\vec{\xi}_n^*=\vec{\xi}_{j^*}\in\mathbb{R}^d\right)$ for the basis function center taken from the training set. The dimensionality of the nonlinear optimization problem is kept low since we are solving only one basis at a time.

2.3 Implementation of the Algorithm

Though our SFA scheme allows the basis center ($\vec{\xi}_n^*$) to be located anywhere in $\mathbb{R}^d$, the practical application to problems with multiple inputs constrains the centers to the set of sample points $\left\{\vec{\xi}_1,\ldots,\vec{\xi}_s\right\}$. At each stage, we determine $\vec{\xi}_n^*$ such that $\left|r_{n-1}\left(\vec{\xi}_n^*\right)\right|=\max\left\{\left|\vec{R}_{n-1}\right|\right\}$. The remaining optimization variable σ_n is continuous and constrained to $0<\sigma_n<1$.

The benefit to using radial basis functions is that in practical applications we can ignore the denominator in the discrete inner product formulation of Eq. (2.10). As a result, the determination of $\vec{\xi}_n^*$ and σ_n requires only

$$\min_{\vec{\xi}_n^*,\sigma_n}\left[-\left\langle \vec{R}_{n-1},\vec{\Phi}_n\right\rangle_D^2\right]. \tag{2.14}$$

The algorithm terminates when $\max\left\{\left|\vec{R}_n\right|\right\} \le \tau$ where τ is the tolerance desired by the user. For binary classification where $u = +1$ or -1, we set $\tau = 0.99$.

In this work, a multi-class classification problem is tackled by applying several binary classifiers in parallel. Approaches to combining binary classifiers include the *one vs. one* combination, the *one vs. all* pairing, and the error-correcting output coding approaches. Beygelzimer, Langford, and Zardony [26] modified the standard *one vs. all* approach and showed that it can also be applied to multi-label classification. Berger [27] explains error-correcting output coding in detail and applies it to the task of document categorization. Masulli and Valentini [28] compared several decomposition schemes for classification.

Rifkin and Klautau [29] compared the *one vs. all* approach to other existing methods and concluded that *one vs. all* classification will produce results as good as any other approach if the underlying classifiers are well tuned. In a standard *one vs. all* formulation applied to a multi-class classification task, M binary classifiers are constructed, where M is the number of classes in the data set.

To implement the SFA algorithm, the user takes the following steps:

1. Choose a class; label all the members of this class as 1 and the members of all the other classes as -1.

2. Initiate the algorithm with $\vec{R}_0 = \left\{u\left(\vec{\xi}_1\right), u\left(\vec{\xi}_2\right), \ldots, u\left(\vec{\xi}_s\right)\right\}$.

3. Search the components of $\vec{R}_{n-1}$ for the maximum magnitude. Record the component index j^*.

4. $\vec{\xi}_n^* = \vec{\xi}_{j^*}$.

5. With $\phi\left(\vec{\xi}, \vec{\xi}_n^*, \sigma_n\right)$ centered at $\vec{\xi}_{j^*}$, calculate b_n from Eq. (2.9) and minimize Eq. (2.10), initializing the optimization parameter with $\sigma_n \cong 0$.

6. Calculate the coefficient c_n from Eq. (2.9).

7. Update the residual $r_n\left(\vec{\xi}\right) = r_{n-1}\left(\vec{\xi}\right) - c_n\left(\phi\left(\vec{\xi}, \vec{\xi}_n^*, \sigma_n\right) + b_n\right)$. Repeat steps 3 to 7 until the termination criterion $\max\left\{\left|\vec{R}_n\right|\right\} \le 0.99$ has been met.

8. Use the constructed binary classifier to predict on the test set.

9. Repeat steps 1 to 8 for remaining classes.

10. Use *one vs. all* scheme to obtain the final prediction on the test set.

Our binary method is linear in storage with respect to s since it needs to store only $s + sd$ vectors to compute the residuals: one vector of length s ($\vec{R}_n$) and s vectors of length d ($\vec{\xi}_1, \vec{\xi}_2, \ldots, \vec{\xi}_s$), where s is the number of samples and d is the number of dimensions. Conventional Support Vector Machines require on the order of s^2 memory [30]. To generate the SFA model requires two vectors of length n ($\{c_1, \ldots, c_n\}$ and $\{b_1, \ldots, b_n\}$) and n vectors of length $d+1$ ($\vec{\xi}_1^*, \vec{\xi}_2^*, \ldots \vec{\xi}_n^*$ and $\sigma_1, \sigma_2, \ldots, \sigma_n$ concatenated as n vectors).

2.4 Testing the Algorithm

Some test cases are presented here to show some typical behaviors of the algorithm. Three functions of two variables each were created to generate a database of 1000 points. The functions are as given:

$$f_1(x_1,x_2) = sign(x_1 x_2) \quad \text{where} \quad \begin{array}{c} -1 \le x_1 \le 1 \\ -1 \le x_2 \le 1, \end{array} \tag{2.15}$$

$$f_2(x_1,x_2) = sign(x_1 |x_2|) \quad \text{where} \quad \begin{array}{c} -1 \le x_1 \le 1 \\ -1 \le x_2 \le 1, \end{array} \tag{2.16}$$

$$f_3(x_1,x_2) = sign(x_1 |x_2|) \quad \text{where} \quad \begin{array}{c} x_1 = \pm 1 \\ -1 \le x_2 \le 1. \end{array} \tag{2.17}$$

The functions all return ± 1.

Each function database was randomly divided into a training set and a test set. A mapping was built based on the training set, then compared against the testing set. For a given training set size, the procedure was repeated 20 times. The training set size was varied from 10% of the data to 90% of the data. The misclassification error trends for each function are plotted in Figure 2.1, Figure 2.5, and Figure 2.9. In these plots, the error bars shown depict the standard deviations of the misclassification error (for 20 permutations). The mapping is shown in the three-dimensional plots in Figure 2.2, Figure 2.6, and Figure 2.10. The actual classification takes the mapping (as shown) and computes the sign of the dependent variable so that the output is discrete.

Based on the mapping (before discretization), input sensitivities are computed from the mean-squared partial derivatives with respect to each input, evaluated at the centers of the basis functions. Figure 2.4, Figure 2.8, and Figure 2.12 plot the input sensitivities, computed from the models which used 50% of the data for training.

2.4.1 Test Function 1

For Function 1, both inputs are expected to be equally significant, since they are weighted and treated equally. From Figure 2.4, this is definitely shown to be the case. It is also comforting that the misclassification error in Figure 2.1 is below 10% as long as at least 100 points are used for training. Figure 2.2 confirms that the mapping (prior to discretization) is accurate in that it resembles a saddle-shape. Figure 2.3 shows that the discretized output maps accurately except at the discontinuities. This is due to the fact that the training points don't exist everywhere along the discontinuous boundaries.

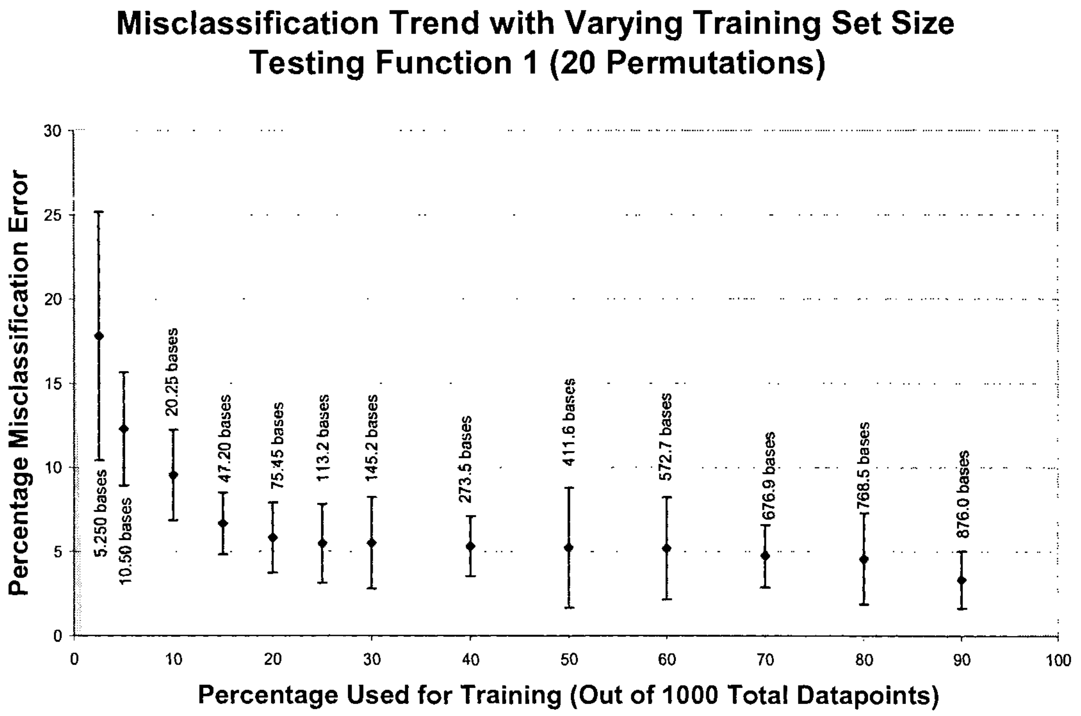

Figure 2.1. Misclassification trend with varying training set size for Function 1. A mapping is built from a varied amount of random training points. This is repeated for 20 different permutations, for a given training set size. The misclassification error is computed by comparing the mapping results

to the remaining data points not used for training. The downward trend suggests an improved accuracy with more training points.

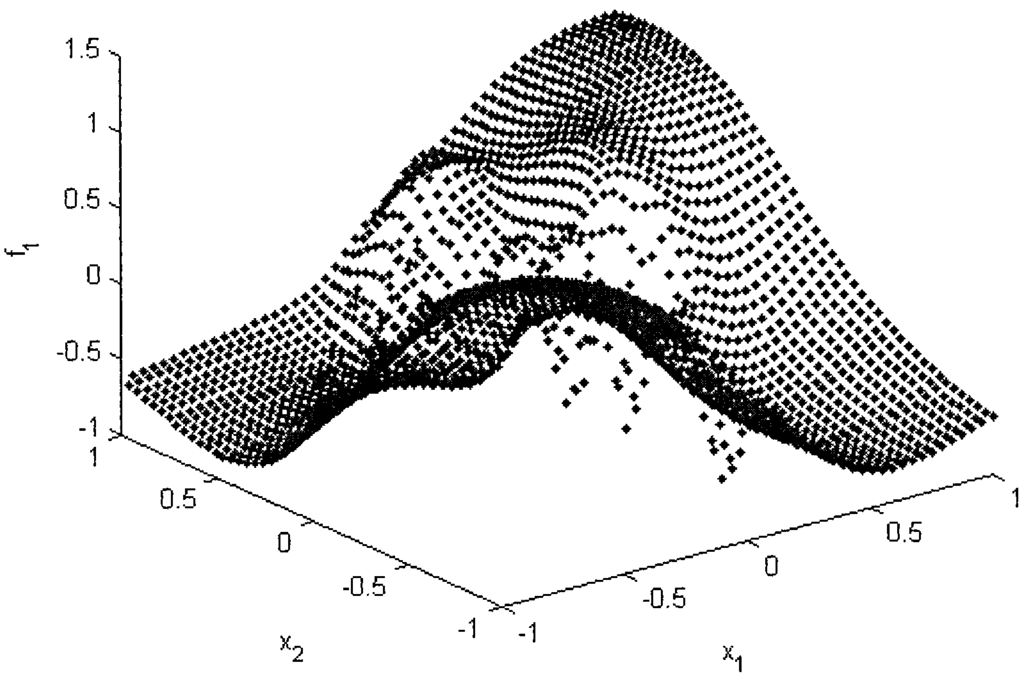

Figure 2.2. Input to output mapping for Function 1. The mapping shown is taken before the output is discretized by the sign function. This particular mapping resembles a saddle-shape, which is a smooth form of the ultimate, discretized version.

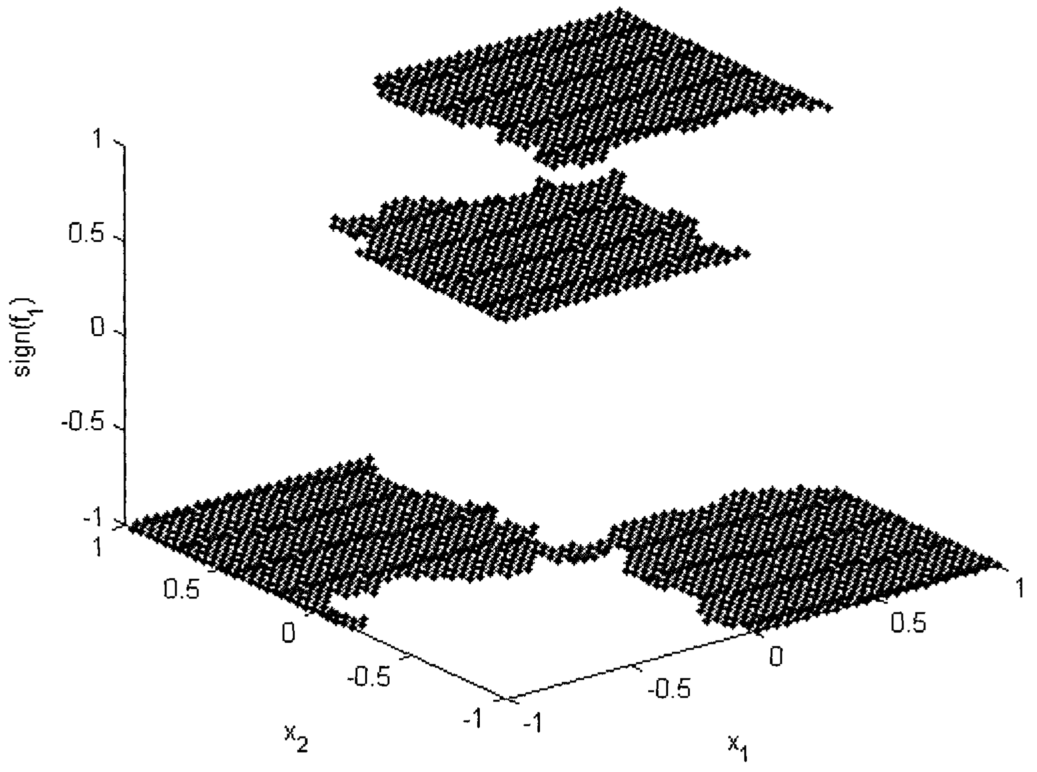

Figure 2.3. Input to output mapping for Function 1 after the output is discretized. Except for the discontinuous boundaries, all the points are mapped correctly.

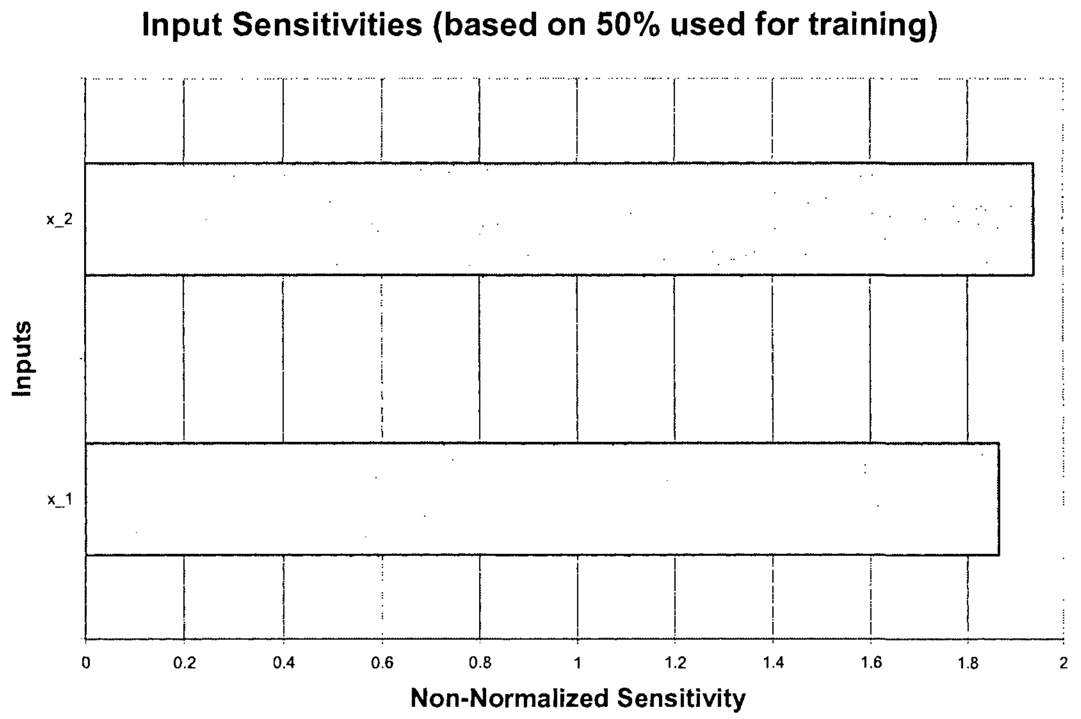

Figure 2.4. Average input sensitivities for Function 1 computed using partial derivatives. Input sensitivities are taken by taking the mean-squared values of the partial derivatives with respect to the input variable, evaluated at the radial basis function centers. These sensitivities were evaluated based on the (pre-discretized) mappings corresponding to 50% used for training. The sensitivities of both inputs for Function 1 are approximately equal, as to be expected.

An Analysis of Variance (ANOVA) test was done to compare against our sensitivity analysis. ANOVA is a test which computes the probability of a null hypothesis. The null hypothesis assumes that variation in a given input has no effect on the mean of the output. In essence, it is another way of quantifying the sensitivity of an input. The p-value for x_1 is 0.7512, and the p-value for x_2 is 0.5075. This means x_1 and x_2 have statistically high probabilities of accepting the null hypothesis. However, the interaction between x_1 and x_2 returns a p-value of 0. In other words, we can conclude with greater than 99.99% confidence that the interaction of the two variables affects the output. This is of course true for Function 1.

2.4.2 Test Function 2

For Function 2, the second input is expected to have a low sensitivity because we take the absolute value of it. Figure 2.8 shows a relative drop in the sensitivity, though it is not insignificantly low. From the mapping in Figure 2.6, it is apparent that there is a minor variation in the output with respect to x_2. The variation is small enough that it does not show up in the discretized output, shown in Figure 2.7. Furthermore, the variation with respect to x_1 is quite shallow in comparison to the discretized version. However, this turns out to be acceptable because the algorithm only needs to learn the sign of the output. The wavy discontinuity in the discretized mapping is the sole source of the error. Again, it is due to the fact that there are not enough training points along the discontinuity. All of this combined reconciles the low misclassification error shown in Figure 2.5.

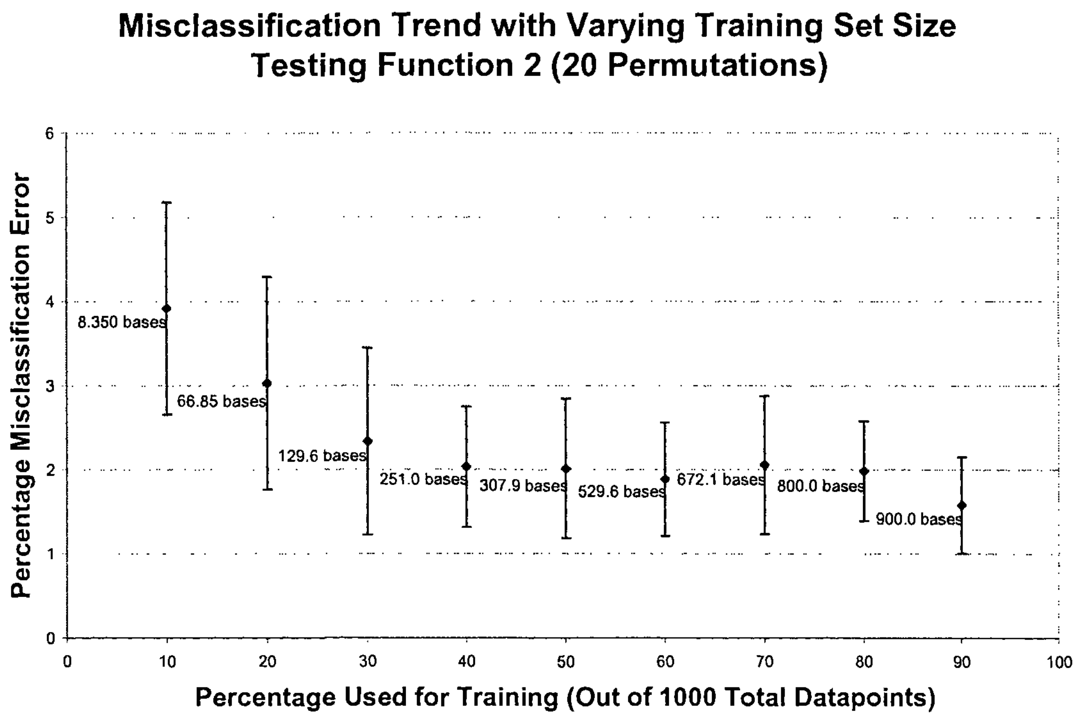

Figure 2.5. Misclassification trend with varying training set size for Function 2.

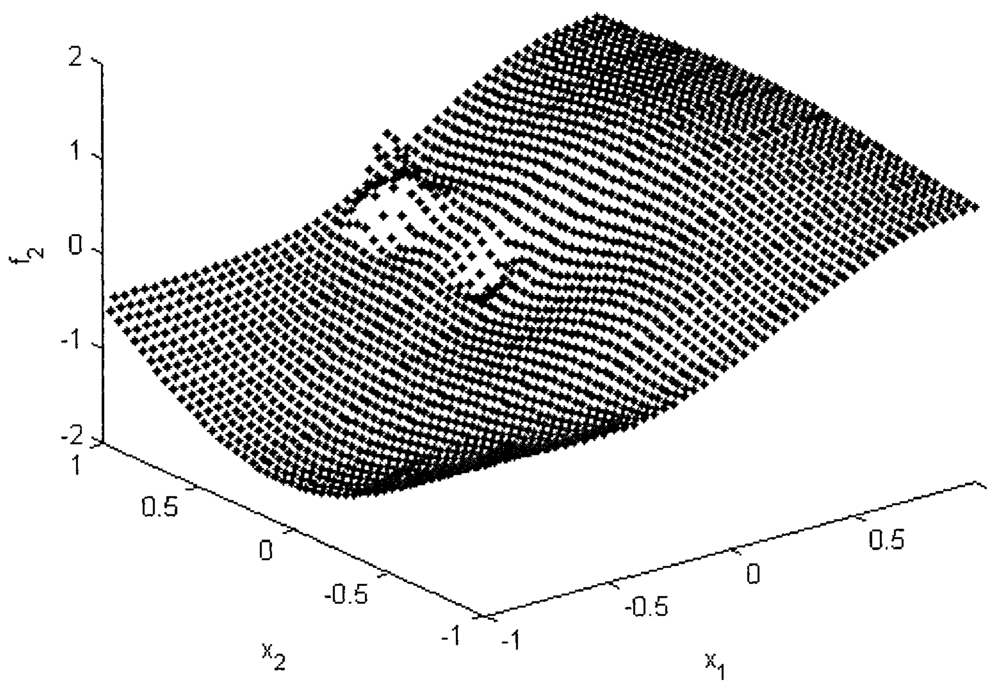

Figure 2.6. Input to output mapping for Function 2. The mapping shown is taken before the output is discretized by the sign function. The ultimate, discretized version should be a step function. This particular mapping is basically a slope, which is a smooth form of a step function.

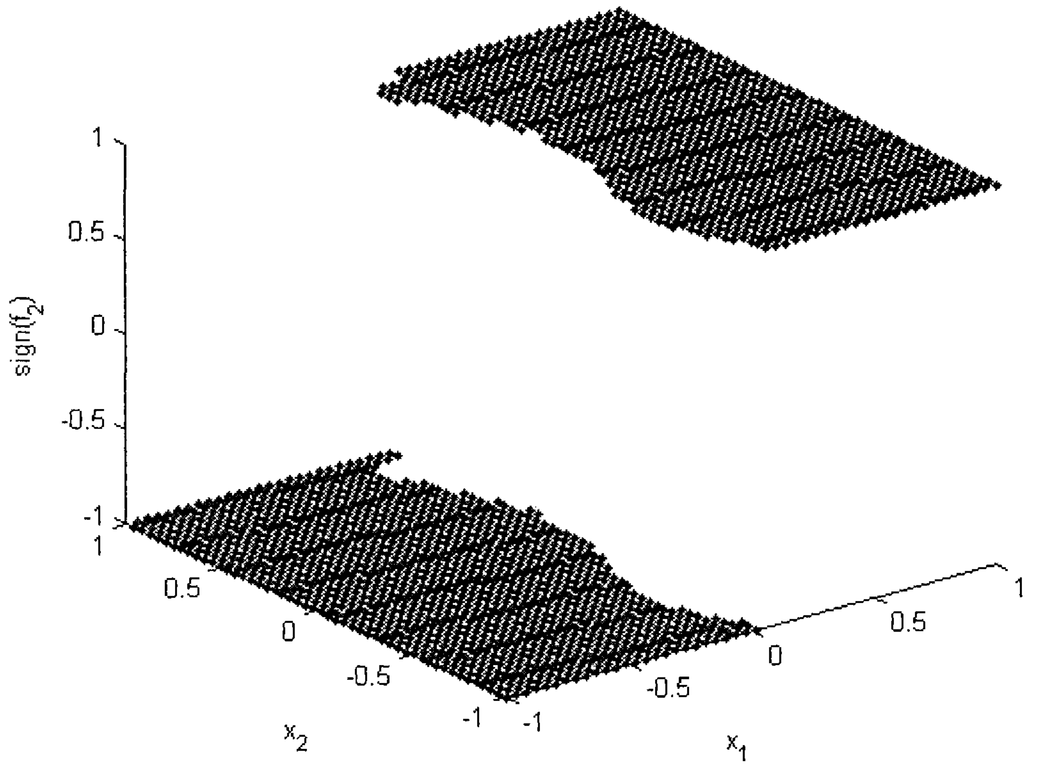

Figure 2.7. Input to output mapping for Function 2 after the output is discretized. Except for the discontinuous boundary, all the points are mapped correctly.

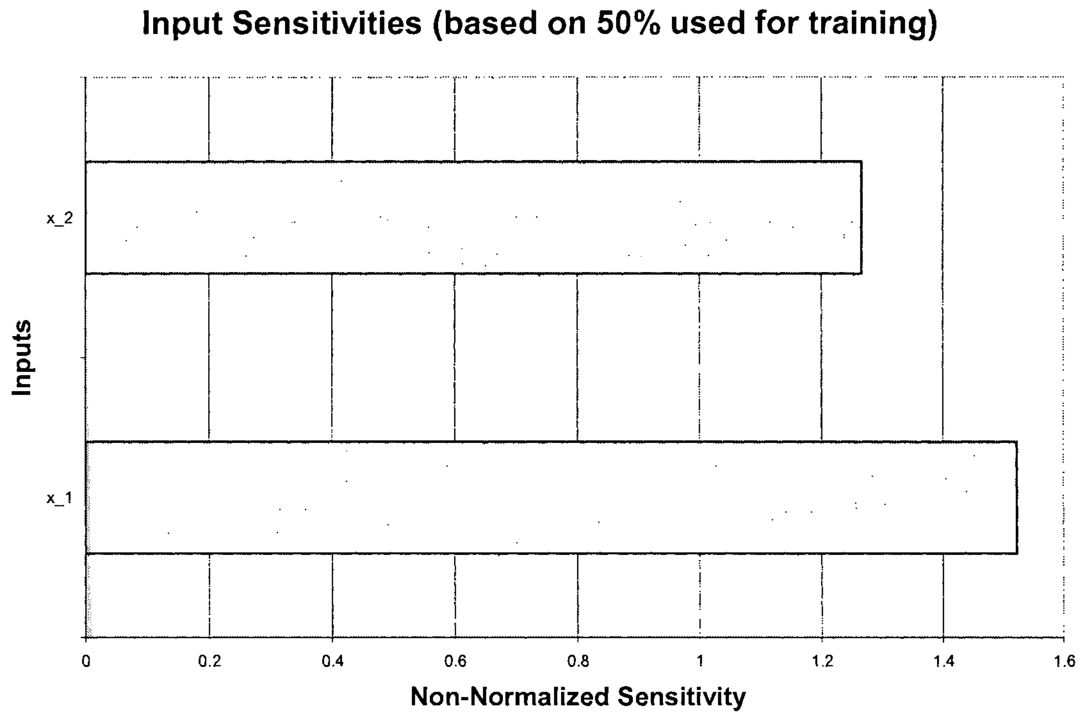

Figure 2.8. Average input sensitivities for Function 2 computed using partial derivatives. The second input's sensitivity is relatively lower than that of the first input, as to be expected.

ANOVA for Function 2 returned $p=0$ for x_1, $p=0.3595$ for x_2, and $p=0.6744$ for the interaction between the two. This means the first input affects the output conclusively, and the second (alone and in conjunction with x_1) has a high probability of accepting the null hypothesis. Again, this is true for Function 2.

2.4.3 Test Function 3

For Function 3, we have modified Function 2 such that the x_1 input has become a binary variable. While this should not affect the output, it clearly has an effect on the sensitivity. In fact, Figure 2.12 shows a dramatic decrease in x_1 sensitivity, so dramatic that it is even less than the x_2 sensitivity. Despite this counter-intuitive result, the misclassification trend in Figure 2.9 shows the best accuracy of the three functions.

Again, we refer to the mappings in Figure 2.10 and Figure 2.11 to reconcile this behavior. Since there is no data in the $-1 < x_1 < 1$ range, the training can not learn about the domain space in that region. The mapping will almost always under-represent the significance of a binary (or discretized) variable. Since the discontinuity exists in between the two valid binary values of x_1 (and far enough away from either), all the points can be mapped correctly for this special case.

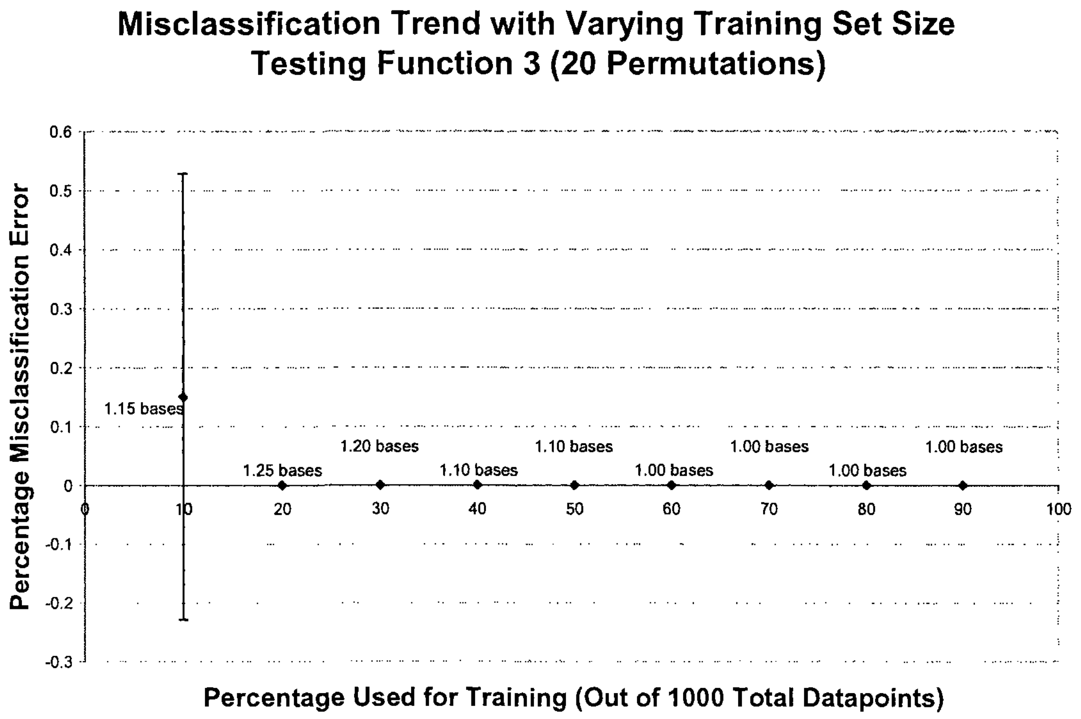

Figure 2.9. Misclassification trend with varying training set size for Function 3.

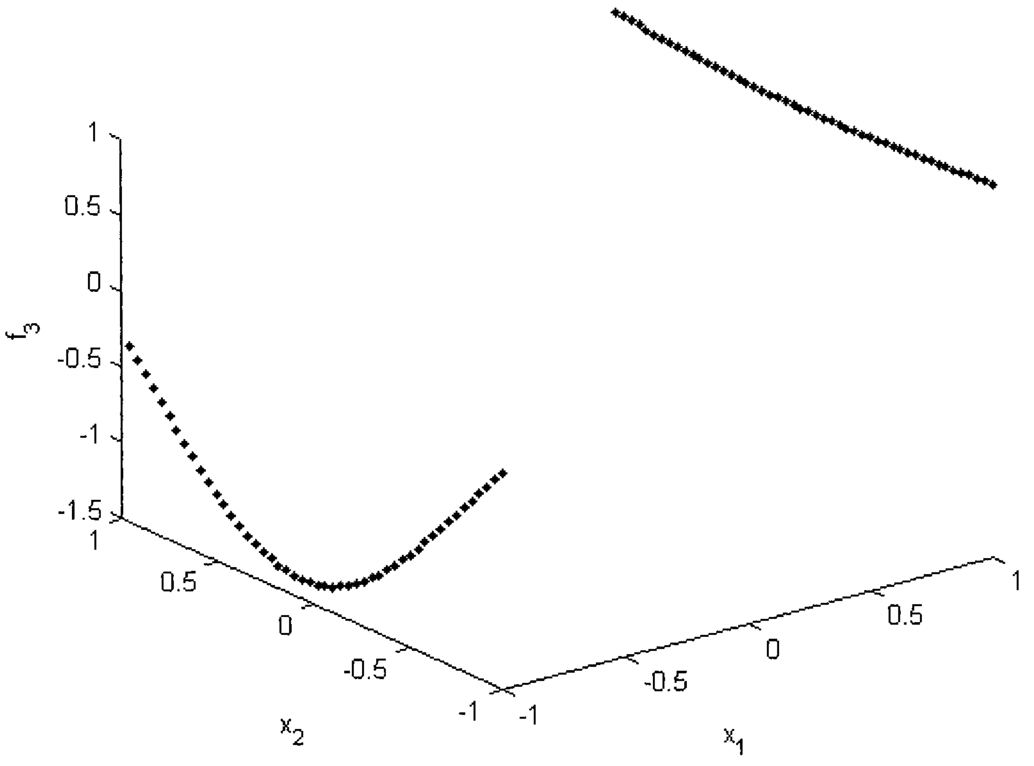

Figure 2.10. Input to output mapping for Function 3. The mapping shown is taken before the output is discretized by the sign function. The ultimate, discretized version should be a row of points at +1 and a row of points at –1. If this particular mapping is discretized, it will match the solution exactly.

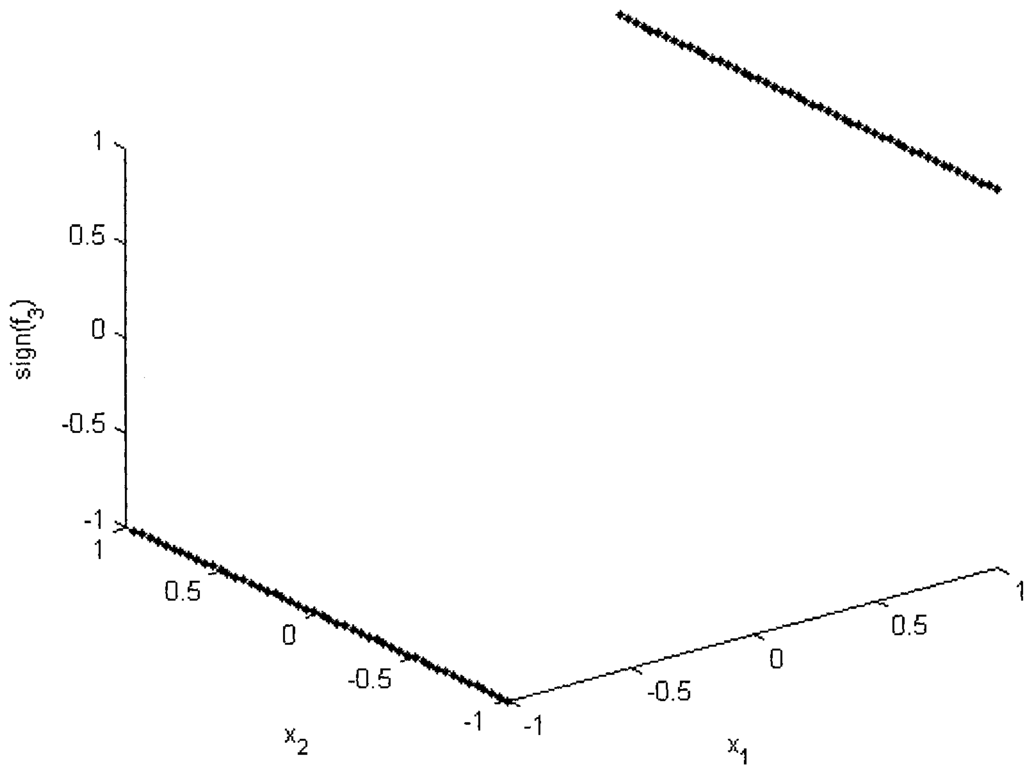

Figure 2.11. Input to output mapping for Function 3 after the output is discretized. Since the first input variable is binary, all the points in the domain are far enough away from the discontinuity so that all the points are mapped correctly.

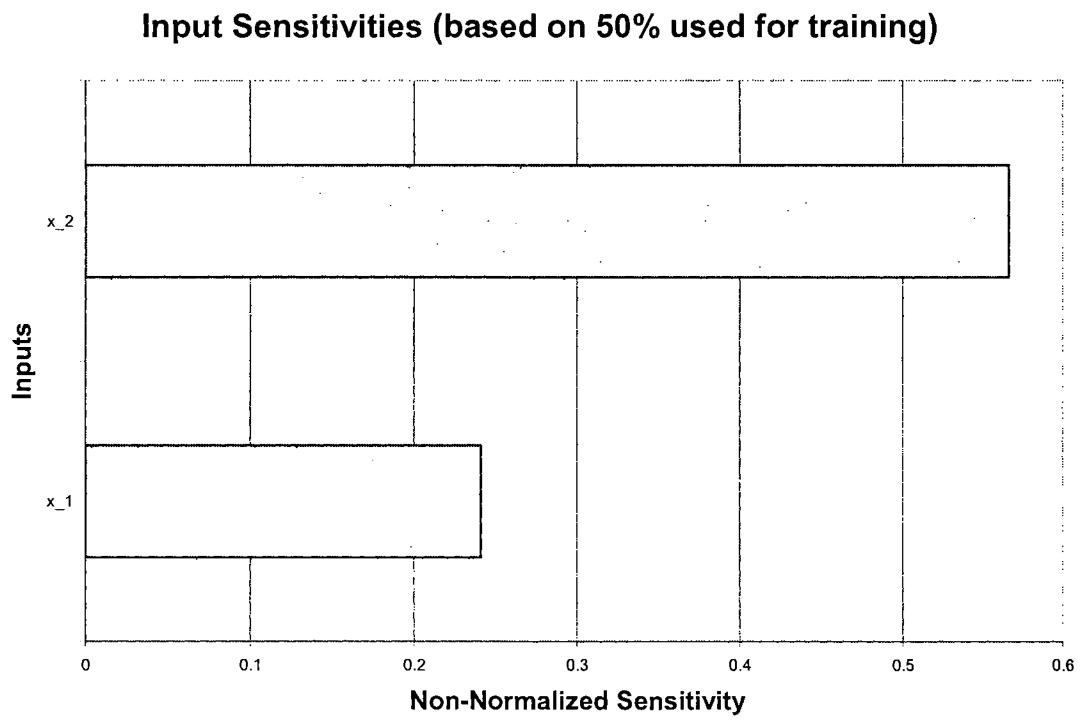

Figure 2.12. Average input sensitivities for Function 3 computed using partial derivatives. The second input's sensitivity is lower than it was in Figure 2.8, but the first input's sensitivity is significantly lower. This unexpected result is due to the binary nature of the input.

ANOVA for Function 3 returned $p=0$ for x_1, $p=1$ for x_2, and $p=0$ for the interaction between the two. This means x_1 and the interaction between x_1 and x_2 affect the output. We can also conclusively say that x_2 alone does not affect the output. This is true for Function 3.

Chapter 3

Approach

We now consider how to apply SFA to composite processing. There are a number of considerations, so the following sections are aimed at addressing each of the concerns. The field of composites is quite large, so clearly defining the problem will help to limit the domain space. For simplicity, this work only takes Single-walled Carbon Nanotube (SWNT) composites into account. A brief literature survey of Multi-walled Carbon Nanotube (MWNT) composites revealed that they would introduce a large subset of input variables that do not correspond to those of SWNT composites [4-6,31-37].

The goal is to determine how various factors will influence the outcome of a Polymer-SWNT composite. Some considerations, therefore, will be what those factors will be and how the "outcome" will be determined and evaluated.

3.1 Composite Processes & Other Factors

The first step to implementing SFA to our problem is to identify the input variables. Below, we will illuminate some of these factors.

3.1.1 CNT

There are many ways that the CNTs can affect the composite. This includes synthesis, received form, chirality, amount, quality, maker, modifications, etc. For simplicity and feasibility sake, we will look at a few of them.

Synthesis – There are four main techniques for CNT synthesis. The first is the Electric Arc Discharge Technique, or Arc Discharge for short. The method was first done by Sumio Iijima in 1991. “An arc discharge is generated between two graphite electrodes placed face to face in the machine’s principal airtight chamber under a partial pressure of helium or argon. The electrical discharge that results brings the temperature up to $6000^{\circ}C$. This is hot enough for the carbon contained in the graphite to sublimate. During sublimation, pressure runs very high, ejecting carbon atoms from the solid and forming a plasma. These atoms head toward colder zones within the chamber, allowing a nanotube deposit to accumulate on the cathode” [1,38].

The second technique is called Laser Ablation. The technique, created by Ting Guo, Richard Smalley, et al, also consists of sublimating graphite in a reduced atmosphere of rare gases. Graphite (and in some cases a graphite-catalyst composite) is vaporized by the laser. The CNTs nucleate in the vapor phase, coalesce, get carried away by flowing argon and condense downstream on a water-cooled collector [38,39].

Another technique is called Chemical Vapor Deposition (CVD). The technique was applied to make CNTs in 1993 by M. José-Yacamán, et al. In this method, a substrate is impregnated with metal catalytic particles. The substrate is heated to $700^{\circ}C$, and two gases are bled into the reactor, including a carbon-containing gas. Decomposition of the carbon-containing gas allows free carbons to grow in a needle-like structure at the catalytic sites. The size of the catalyst particles determines the diameter of the resulting CNTs [38,40].

The last technique is known as High Pressure Carbon Monoxide (HiPco). The technique was first investigated and optimized by Michael Bronikowski, Richard Smalley, et al in 2001. $Fe(CO)_5$ particles are injected to a stream of high temperature and high pressure Carbon Monoxide (CO) gas. Upon heating, the $Fe(CO)_5$ decomposes into iron particles which condense into clusters. The clusters serve as catalysts upon which the CNTs nucleate and grow [41,42].

Received Forms of CNTs – CNTs can be found in various macroscopic forms. Common dry aggregates such as powder and pearls differentiate themselves from as-produced raw material (commonly referred to as fluff) through higher densities. The different forms are the same on the nanoscopic level and the attributes are unaltered. The different forms are simply different post-synthesis processes which may be beneficial for different applications.

Purification – The various synthesis procedures often leave carbon nanoparticles, residual catalysts, and other unwanted species. Purification is a process which separates out those particles with minimal loss of CNTs. The intrinsic properties of CNTs can only be studied if these impurities are removed [42,43]. There are various methods of purification, but they will not be discussed in this thesis.

Functionalization – "Functionalization is the process by which chemical functional groups are introduced to the surface of a material. Functionalization can be used to modify the interface between the environment and the outer wall of CNTs. This would, for example, modify the solubility and facilitate the dispersion of a given medium" [38]. The atomically smooth and nonreactive surface of CNTs limits the load transfer from polymer to CNT. With little to no interfacial interaction, the CNTs are inclined to pull out of the polymer. All known forms of SWNTs are insoluble in organic solvents. Organic functional groups can be covalently attached to CNT surfaces, opening the window to a wide range of possibilities [38,44,45].

Weight Percent – The weight percent is a measure of the amount of CNTs present in a composite. The higher the weight percent, the more CNTs there are. Naturally, this affects the dispersion and other properties of CNT composites.

3.1.2 Polymers

The polymer (also referred to as the matrix) is the material that surrounds the CNTs in the composite. "A polymeric solid material may be considered to be one that contains many chemically bonded parts or units which themselves are bonded together to form a solid" [41]. This research focuses on two main types of polymers: thermoplastics and thermosetting plastics (thermosets).

Thermoplastics – "Thermoplastics require heat to make them formable and after cooling, retain the shape they were formed into. These materials can be reheated and reformed into new shapes a number of times without significant change in their properties. Most thermoplastics consist of very long main chains of carbon atoms covalently bonded together. Sometimes nitrogen, oxygen, or sulfur atoms are also covalently bonded in the main molecular chain. Pendant atoms or groups of atoms are covalently bonded to the main-chain atoms. In thermoplastics the long molecular chains are bonded to each other by secondary bonds" [41]. Some examples of thermoplastics include polyethylene, polypropylene, polystyrene, ABS, and polymethyl methacrylate. CNTs are generally dispersed in thermoplastics via chemical methods.

Thermosetting plastics (thermosets) – "Thermosetting plastics formed into a permanent shape and cured or 'set' by a chemical reaction cannot be remelted and reformed into another shape but degradate or decompose upon being heated to too high a temperature.... Most thermosetting plastics consist of a network of carbon atoms covalently bonded together to form a rigid solid. Sometimes nitrogen, oxygen, sulfur, or other atoms are also covalently bonded into a thermoset network structure" [41]. Some thermosets include epoxies and polyester. CNTs are generally dispersed into thermosets via mechanical methods.

3.1.3 Dispersion Methods

There are many types of dispersion methods. For discussion purposes, we will categorize them into two main classes: chemical and mechanical methods.

Chemical Methods – Chemical methods include incipient wetting and in situ polymerization. Incipient wetting is a method which mixes the CNTs and polymer in a suitable solvent before evaporating the solvent to form the composite. In this method, agitation of the CNTs in the solvent serves to de-agglomerate and disperse the CNTs. Agitation is generally provided by magnetic stirring, shear mixing, or most commonly sonication. "The solution is combined with a polymer, and the solvent is evaporated using an oil bath. After the solvent is removed, an additional furnace heating step is performed to ensure that all of the solvent is removed. Polymer powders are particularly well suited to this processing step because they have more surface area than pellets have. The solvent chosen is specific to the polymer being used. It is desirable for the boiling point of the solvent to be lower than the polymer's melting temperature to allow coating of the polymer particles" [7,8].

In situ polymerization has been suggested as an ideal way to make of perfect dispersion of CNTs into polymer matrices. The process works by grafting polymer macromolecules onto the walls of the CNTs. The objective is two-fold. First, the CNTs are prohibited from agglomerating with each other because there are long polymer chains attached to the sides. Secondly, the CNTs are automatically incorporated into the matrix because the matrix material doesn't see the CNTs, only the polymer chains around them [8,9,46].

Mechanical Methods – We will discuss two types of mechanical mixing methods: sonication and high shear mixing.

Sonication is a process which exposes the material to ultrasound, in an attempt to disperse CNTs in the liquid media. Air layers and gas bubbles tend to be trapped within the liquid composite, and the ultrasonic waves are reflected off of these air-liquid interfaces. In general, the waves are prevented from hitting the CNT surfaces. The sonic waves help to break up bundles and increase CNT surface area. The increased surface area improves the wetting by making the CNT surfaces more accessible to the solvent [10,47].

High shear mixing (or sometimes melt mixing) involves a pair of high shear rotor blades. The temperature of the mixing chamber is increased such that the polymer becomes molten. High shear mixing is most suitable for thermoplastic polymers because they can be softened under high heat. Under the high shear mixing, the CNT disentanglement is facilitated and secondary agglomeration is prevented. At low shear, re-agglomeration of well-dispersed CNTs can be observed [8,11,48]. Another type of mechanical mixing which we have considered is called homogenization. A homogenizer is essentially a high shear mixer except it is done at room temperature.

3.2 Input Selection

Based on a literature survey of polymer-SWNT composite research and the summary we have provided above, we have chosen a list of 20 input variables, shown in Table 3.1 [7,12-15,48-64].

Input Number	Input Variable	Sample Input Values
1	How are CNTs made?	0 – Laser Ablation
		1 – HiPco
		2 – CVD
		3 – Arc Discharge
2	How are the CNTs received?	0 – Pearls
		1 – Powder
		2 – Fluff
3	How are CNTs modified?	0 – not modified
		1 – purified
		2 – functionalized, fluorination
		Etc. (other functional groups)
4	Are CNTs shortened?	0 – No
		1 – Yes
5	Are solvents added? If so, what solvent?	0 – No
		1 – Yes, Water
		2 – Yes, Chloroform
		Etc. (other solvents)
6	What is the weight percent of CNTs?	Any value (percentage)
7	What is the polymer used?	0 – Polypropylene
		1 – Low Density Polyethylene
		2 – Medium Density Polyethylene
		Etc. (other polymers)
8	Is incipient wetting done?	0 – No
		1 – Yes
9	How long is it mixed manually?	Any value (minutes)
10	How long is it mixed with a magnetic stir bar?	Any value (minutes)
11	How long is it homogenized?	Any value (minutes)
12	How long is it sonicated?	Any value (minutes)
13	What is the starting high shear mixing temperature?	Any value (Celsius)
14	What is the stopping high shear mixing temperature?	Any value (Celsius)

15	What is the high shear mixing time?	Any value (minutes)
16	What is the high shear mixing rate?	Any value (RPM)
17	Is stretching done?	0 – No
		1 – Yes
18	Is in situ polymerization done?	0 – No
		1 – Yes
19	Is the composite made in a magnetic field?	0 – No
		1 – Yes
20	Are the CNTs sprayed onto the polymer?	0 – No
		1 – Yes

Table 3.1. Input Variables Selected.

As more data is acquired, this list may be modified. Clearly, the accepted values for Inputs 3, 5, and 7 is a continually growing list. The last several inputs (17-20) are amendments due to new data that provided these methods. However, not every input variable is so easily amended. Some processing methods include factors that go far beyond "time." For example, Input 12 asks, "how long is it sonicated?" Along with sonication time, other factors such as sonication power and sonicator type exist. Unfortunately, many journal articles neglect to report some of this information. In order to move forward, we must accept the fact that some information is unobtainable. For this reason, sonication power and sonicator type are not included as input variables. The same discussion can be applied to several of the other processing methods.

Inputs 13 and 14 provide another challenge. The accepted values for these inputs are temperatures. Let us take Experimentalist A, for example. Experimentalist A chooses not to mix Composite A by high shear mixing. The question then becomes: how did Experimentalist A perform their high shear mixing? The human response is obvious: "they didn't." However, the algorithm requires a response to each of those input variables. Clearly, we could say Experimentalist A performed high shear mixing with "time = 0" and "rate = 0." But what of the temperature? There is no "null" response for temperature. Admittedly, this is one major flaw in this approach. We have assumed that if we choose a "normal" or "average" answer that it would not affect the outcome significantly. In other words, the inputs for a "null" response might be given as in Table 3.2.

Input Number	Input Variable	Input Value
13	What is the starting high shear mixing temperature?	27°C
14	What is the stopping high shear mixing temperature?	27°C
15	What is the high shear mixing time?	0 minutes
16	What is the high shear mixing rate?	0 RPM

Table 3.2. Example of Assigning Input Values to Non-Applicable Input Variables.

3.3 Output Selection

There are many ways to assess the final product of a composite. With mechanical applications, one could look at the enhancement in tensile strength, Young's modulus, etc. With other applications, one could look at thermal conductivity, electrical conductivity, or viscosity. In fact, all of these could be used simultaneously as the output. However, a value for the output must be given in order for the data to be used. If data is obtained from a journal article which focuses on mechanical properties, then the thermal, electrical, and rheological outputs may be missing. Therefore, in order to maximize the amount of data that can be acquired, the output selection must be a generic one.

CNT dispersion is typically one assessment of how successful a composite process is. This is due to the fact that material property enhancement is generally accomplished by achieving a homogenous dispersion of CNTs within the matrix. Therefore, dispersion is a general property that can (and should) be reported by all composite experimentalists – regardless of the ultimate application in mind. For this project, we have chosen to use dispersion as the output.

Another complication arises due to the fact that CNT dispersion is not quantifiable yet. To date, CNT dispersion is always reported in a qualitative manner based upon a Scanning Electron Microscope (SEM) image of the sample. For our database, we have created a rating system to judge the dispersion (Table 3.3). Each rating is treated as a class, and therefore we have transformed our problem into a classification problem, as discussed in the previous chapter. Each sample is given a rating based on a corresponding SEM image.

Output value	Qualitative Explanation
1	Poorly distributed entangled ropes
2	Well distributed entangled ropes
3	Poorly distributed ropes
4	Well distributed ropes
5	Poorly distributed single tubes
6	Well distributed single tubes

Table 3.3. Dispersion as the Output.

3.4 Acquiring Data

Data has been acquired from a number of sources [7,12-15,48-64]. Sources include dissertations, journal articles, and unpublished results from associates. However, with each source, the data must be methodically extracted to fill in all the input and output values. Unfortunately, quite often various pieces of information will be missing from a full data entry. In those cases, authors must be contacted to obtain those missing values. Successful correspondence rate is less than 50%. To date, 134 entries have been added to the database. Regrettably, only 48 of those 134 are complete. The results in the next chapter are based upon those 48 entries.

Also, occasionally an experimental procedure is so radical that it simply will not work with the set of inputs that have been selected. Much like the discussion of MWNTs from the beginning of this chapter, there are even polymer-SWNT composite processes that don’t work well. In those cases, we must accept a loss, and disregard that data.

Chapter 4

Results

4.1 Training and Testing

After the database is acquired, it is divided into two groups: a training set and a testing set. A percentage of the total number of entries is allocated to training. The points are randomly selected from the whole database, and the remaining points are allocated to testing.

The training points are fed to the algorithm so that the computer can learn about the domain space. A mapping is constructed as described in Chapter 2. Assuming the tolerance is reached, the mapping matches the training points exactly. The remaining points, which comprise the testing set, are compared against the mapping. The number of points unsuccessfully mapped is accumulated for "misclassification error." The training and testing procedure is repeated 50 times with a random sampling permutation each time.

4.2 Model Accuracy

The percentage misclassification error is plotted against the number of data points used for training in Figure 4.1. The error bars depict the standard deviation of the misclassification error, after 50 permutations. One can clearly see a downward trend with increasing size of the training set, suggesting improved models with more data. The slight plateau seen at the end is also an expected outcome as seen from our tests in Section 2.4. However, the misclassification error only goes down to about 40%. When SFA was applied to other engineering problems with much larger databases, the error was seen to drop below 10%. Moreover, with the large simulated data in our tests in Section 2.4, we saw error down by an order of magnitude. This is a strong (and optimistic) suggestion that the current database should be increased to see significant improvements in accuracy.

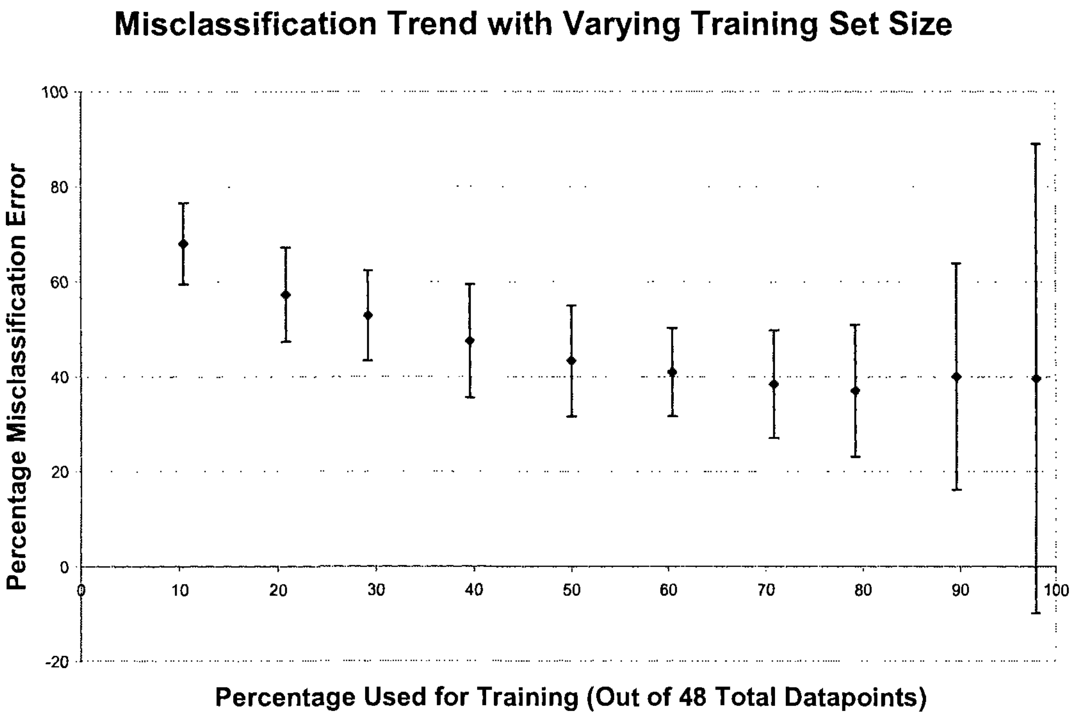

Figure 4.1. Misclassification trend with varying training set size. A mapping is built from a varied amount of random training points. This is repeated for 50 different permutations, for a given training set size. The downward trend suggests an improved accuracy with more training points. The increasing error bars, which denote the standard deviation, are a statistical side-effect due to decreasing number of testing points.

Of course, one can also clearly note that the error bars increase dramatically as the training set increases. To understand this trend, one must realize the varying size of the testing set. For a training set size of 5 points, the test set is 43 points. Testing on so many points improves the confidence of the result. Conversely, for 43 points used for training, only 5 data points are used for testing. With such a small sample size, the confidence of the result is very poor. This is bolstered by the fact that the quality of the model is less than perfect (40-70% misclassification error). Therefore, the increasing error bars is a logical outcome.

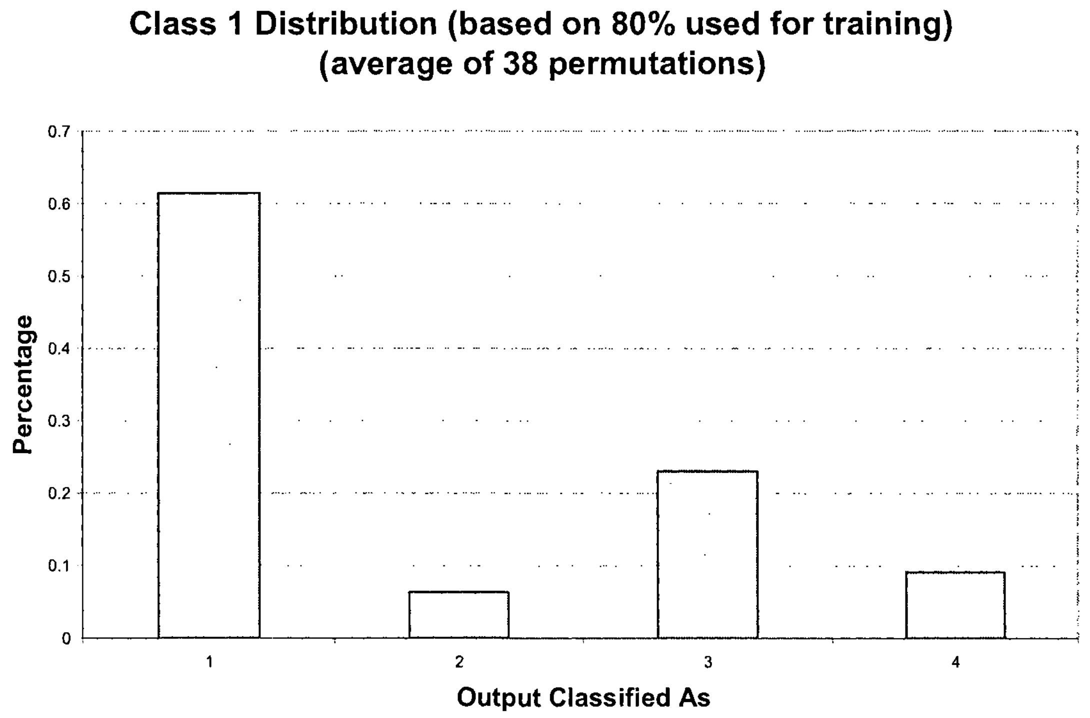

Figure 4.2. Distribution of classification for Class 1. The percentage of test points which were classified to the various output classes where the correct classification was Class 1. The results plotted are an average of 38 random permutations where 80% of the total data was assigned to training.

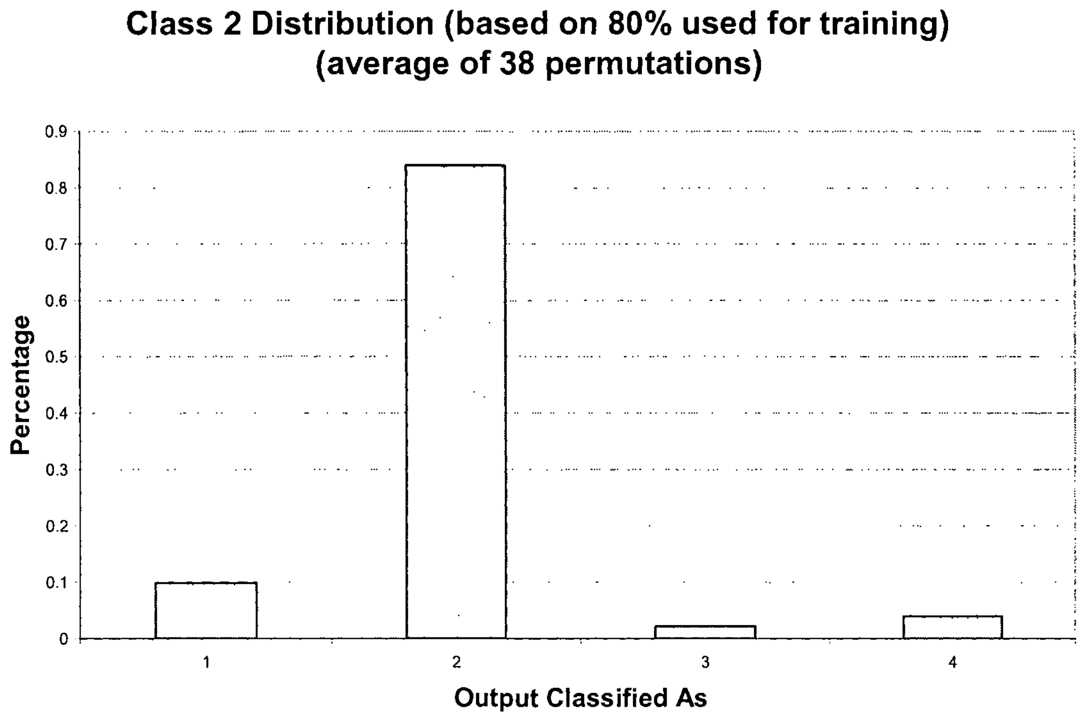

Figure 4.3. Distribution of classification for Class 2.

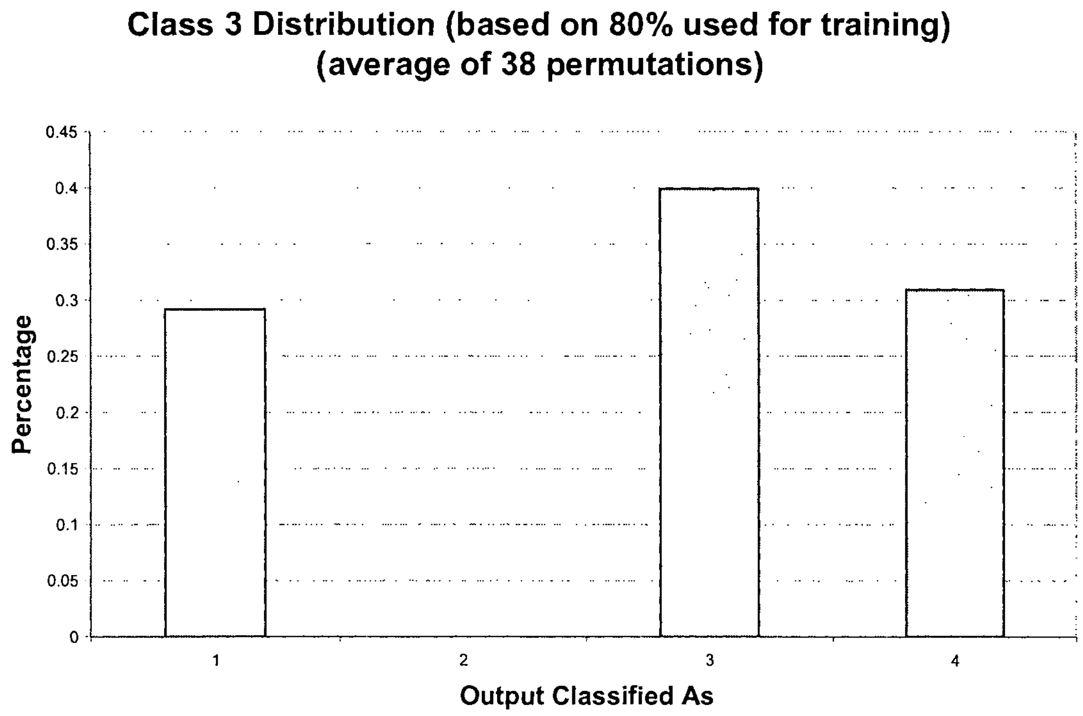

Figure 4.4. Distribution of classification for Class 3.

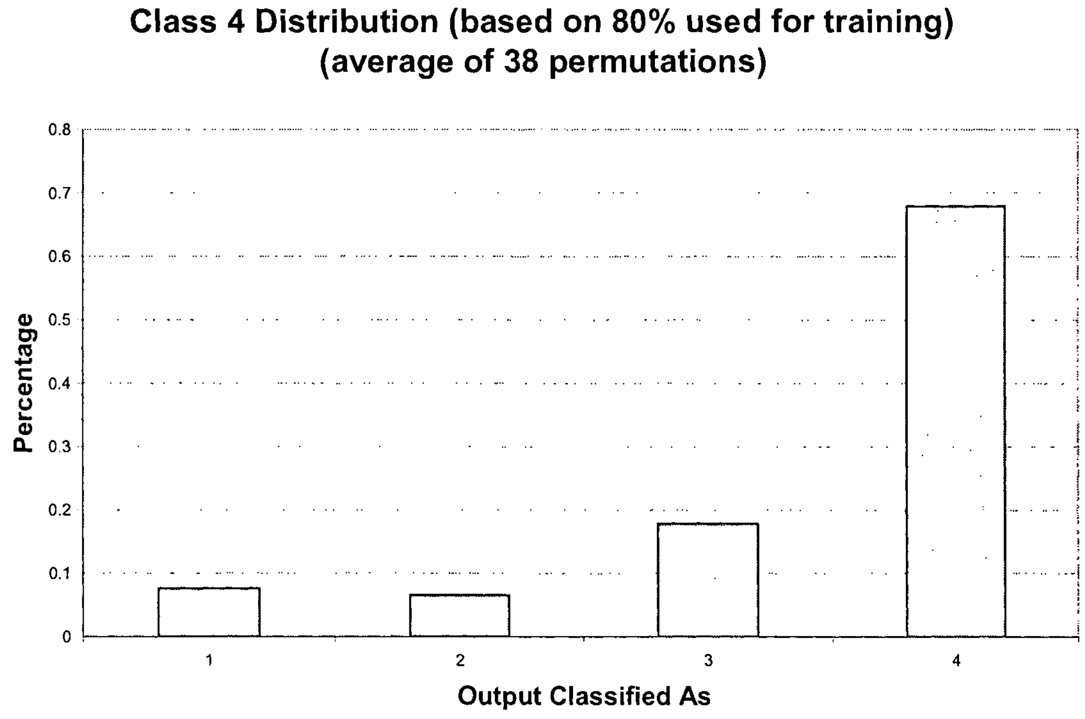

Figure 4.5. Distribution of classification for Class 4.

Figure 4.2 to Figure 4.5 show the confusion matrix in the form of four histogram plots. The confusion matrix contains information about how the actual data is classified versus how the modeled data is classified. For example, Figure 4.2 contains all the test points which have an actual output value of 1. The bars show the percentages which were correctly classified as 1, and the percentages which were incorrectly classified as 2, 3, and 4. Figure 4.2, Figure 4.3, and Figure 4.5 show that a significant percentage of those data were correctly classified and that most of the misclassifications weren't off by a substantial margin. Figure 4.4 however, clearly shows some confusion for Class 3. It is probable that many data points with Class 3 share similar input values as those with Classes 1 and 4.

4.3 Input Sensitivities

Another goal of this project is to find which inputs the target function is most sensitive to. After a mapping is constructed, the input sensitivities can be computed by taking the partial derivatives of the approximating function with respect to each input. The input sensitivities were determined by summing the squares of the derivatives over the bases used to construct the approximating function. This was done once for each binary classifier and an average was taken to get the final input sensitivities. The sensitivities are plotted in Figure 4.2, normalized by the least sensitive input, "Incipient Wetting". It appears that the most sensitive inputs are "CNT Weight Percent," "Sonication Time," "CNT Modification," and "High Shear Time."

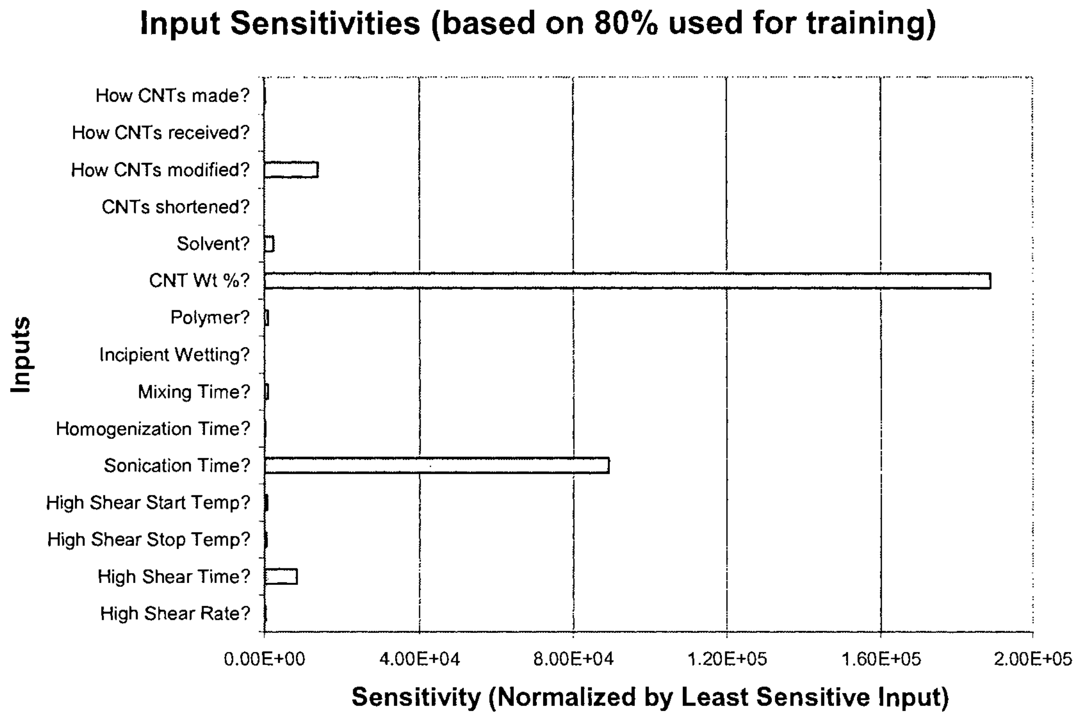

Figure 4.6. Average input sensitivities computed using partial derivatives. Input sensitivities are taken by taking the mean-squared values of the partial derivatives with respect to the input variable, evaluated at the radial basis function centers. These sensitivities were evaluated based on the mappings corresponding to 70% used for training. All input variables are included in the modeling. The sensitivities were normalized by the least sensitive input, "Incipient wetting." Four obvious input variables stand out as important factors: "CNT modification," "CNT weight percent," "Sonication time," and "High Shear Mixing time."

In order to discern the least sensitive inputs, Figure 4.7 shows the input sensitivities on a logarithmic scale. From this figure, the least sensitive inputs are "Incipient Wetting," "CNTs shortened?," "How CNTs received?," and "Homogenization Time."

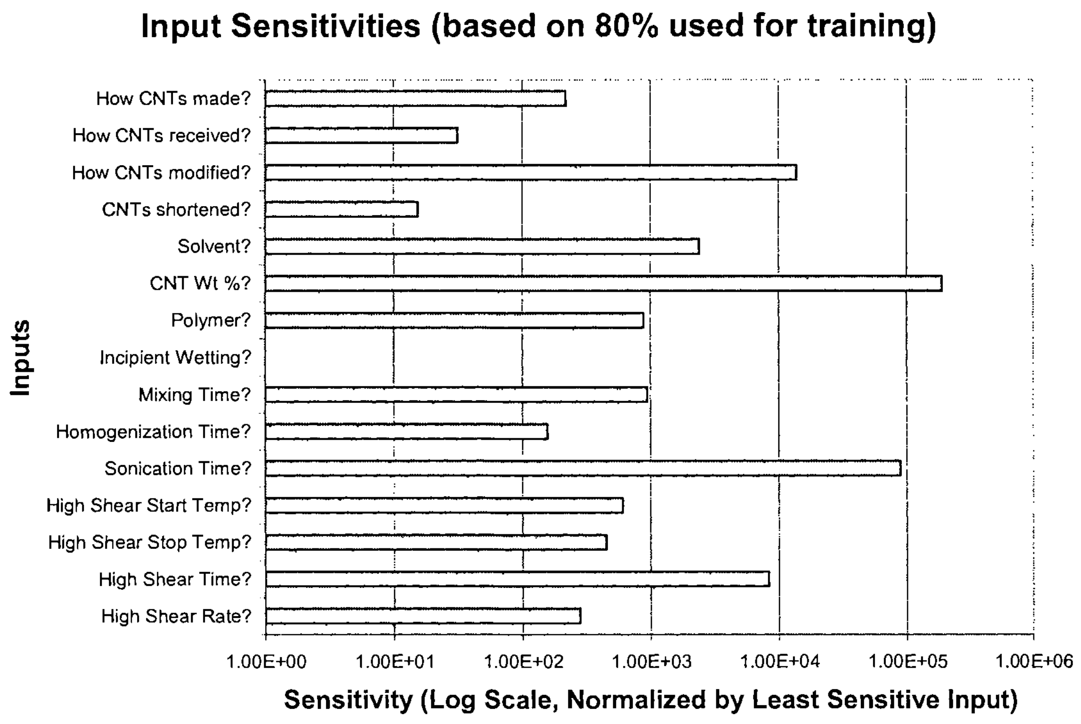

Figure 4.7. Average input sensitivities replotted on logarithmic scale. This is the same plot as Figure 4.6 except the x-axis scale is logarithmic in order to show the lesser sensitive inputs. Here one can see "Form of received CNTs," "CNTs shortened?," "Incipient Wetting," and "Homogenization Time" are the least sensitive inputs.

Upon further evaluation, the sensitivities are scaled by the range of each input variable. This serves to non-dimensionalize the sensitivities so that they can be compared more directly. Figure 4.8 and Figure 4.9 show the scaled results plotted on the linear and logarithmic scales, respectively.

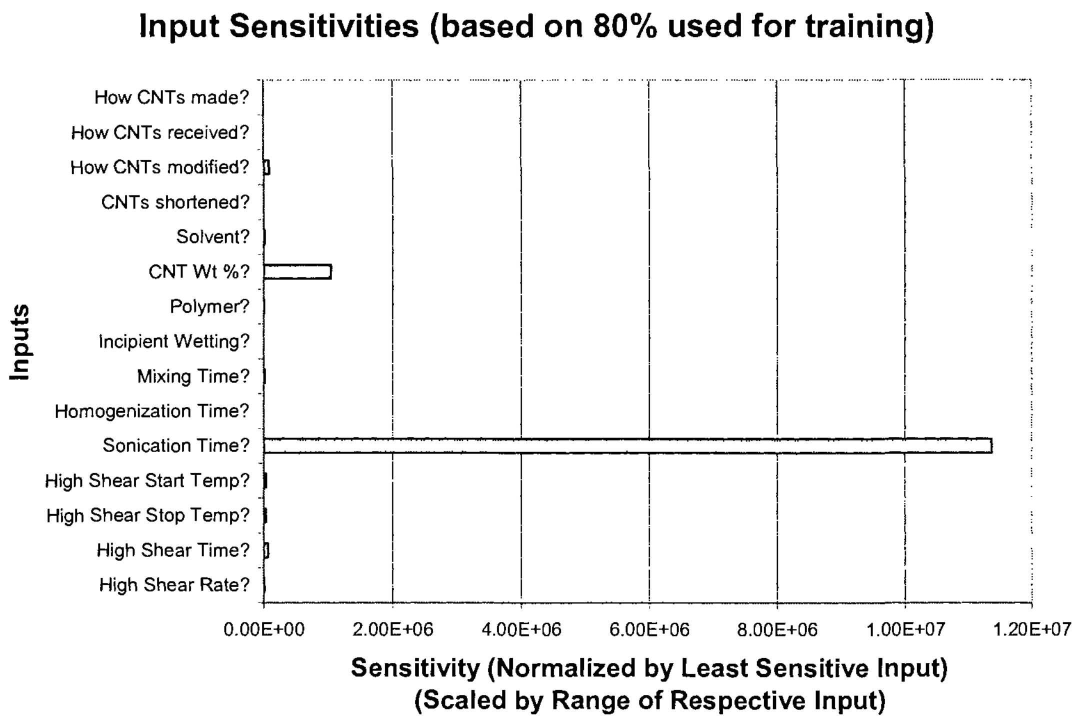

Figure 4.8. Average input sensitivities scaled by range of respective input. Values from Figure 4.6 were scaled by multiplying each sensitivity by the total range of the respective input. Plotted on a linear scale, the figure shows the same four inputs as being the most sensitive, but the order is slightly changed.

According to the scaled figures, the most sensitive inputs remain the same, but in a slightly different order. The least sensitive inputs are “Incipient Wetting?,” “CNTs shortened?,” “How CNTs received?,” and “How CNTs made?”

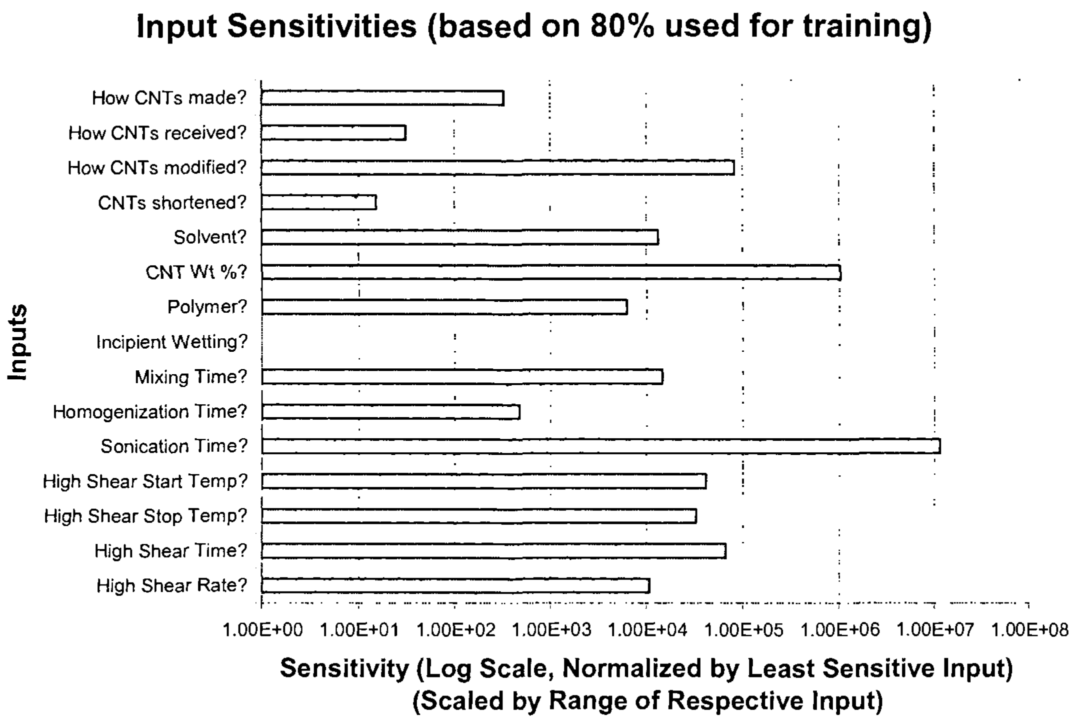

Figure 4.9. Average input sensitivities replotted on log scale, scaled by range of respective input. This is the same plot as Figure 4.8 except the x-axis scale is logarithmic in order to show the lesser sensitive inputs. Here one can see "CNT Synthesis," "Form of Received CNTs," "CNTs Shortened?," and "Incipient Wetting" are the least sensitive inputs.

The following sub-sections are devoted to providing an explanation for these sensitivity results. However, it should be understood that the sensitivities can be misleading in certain situations. When the variation of an input is small, then the domain space with respect to that input is not well-explored. This is well demonstrated in the examples in Section 2.4, for Case 3 where one of the inputs is binary. Certainly with only 48 data entries, there will be many inputs that only reveal a small range of the domain to the computer. Some of these examples will be addressed in the following sub-sections.

4.3.1 CNT Weight Percent

It is well understood that CNT weight percent is important for dispersion. Of course if there are fewer CNTs inside the polymer, it is expected that they would be easier to disperse. Likewise, many CNTs would tend to agglomerate, giving a very poor dispersion. Moreover, as per the argument given before this sub-section, the data acquired contains much variation in the weight percent, so the full extent of its effect on dispersion is more likely to be apparent in our model.

4.3.2 Sonication

Similar to weight percent, it is very reasonable that sonication would have a large effect on dispersion. Sonication is used specifically to break bundles and improve dispersion. So this result is as expected. Again, the data acquired also contains a lot of variability in the sonication input, so its effect is well-represented in our model.

4.3.3 CNT Modification

Functionalization is also expected to play a significant role in CNT dispersion. Functionalization helps to bond CNTs to the matrix (covalently or otherwise). This serves to prevent movement and aggregation of CNTs. However, the choice of functionalization is important and depends on the choice of polymer. A certain functionalizaion may help significantly with a certain polymer, but may be completely useless for another type of polymer. From our acquired data, the CNT modification input spans a wide range as well, but the spread within the whole domain for a given modification remains sparse. It is likely that given more data, CNT modification could prove to be even more important than shown here.

4.3.4 High Shear Mixing

High Shear Mixing is a method of mechanical mixing. The main purpose of this method is also to improve dispersion. However, the results here tend to confuse the issue. On one hand, High Shear Mixing Time is shown to be an important factor while High Shear Mixing Rate seems to be one of the least important factors. One must note that the "Time" input ranges from 10 minutes to 15 minutes (ignoring the null values). On the other hand, the "Rate" input ranges from 60 RPM to 75 RPM (also ignoring the null values), where the majority take the 75 value. The maximum for the "Time" input is 50% larger than the minimum value, while the maximum for the "Rate" input is only 25% larger than the minimum value. Therefore, the "Time" variable covers a wider range than the "Rate" input. Furthermore, the spread of the "Rate" variable is unbalanced within the domain, since the majority of the values are either null or 75 RPM. It is essentially a binary variable.

4.3.5 Incipient Wetting

Incipient Wetting is a chemical method of dispersion. It is specifically designed to increase dispersion. However, we found it to be the least sensitive input. The most reasonable explanation is simply that it is a binary variable. Moreover, with the data provided, incipient wetting was performed whenever the materials called for it. When thermoplastics were used, incipient wetting were done. When thermosets were used, mechanical mixing methods were done. Therefore, it is quite possible that, had incipient wetting been done when it was not called for, it would show up in the data and therefore be apparent in the results.

4.3.6 Shortened CNTs

Shortened CNTs should certainly play a role in dispersion. When CNTs are shortened, there is less tube-tube interaction, so there will be fewer agglomerates. It is expected that shorter tubes would therefore be significantly easier to disperse. However, the results here show a very limited result. There are two very simple reasons. First, the input is a binary variable. There is virtually no variability with binary variables. The computer is unable to explore the space between the minimum and maximum of this variable. The second reason is apparent upon inspection of the database. Of the 48 data entries, only two provide non-null results. This also reduces the variability of the input.

4.3.7 Form of Received CNTs

The received form of the CNTs basically suggests the density of the starting material. The density is essentially the amount of aggregation (of the starting material). Again, this is expected to be an important factor, yet the results show otherwise. One physical reason may be that the processes that were used to disperse the denser CNTs happened to be more successful. However, a more reasonable explanation goes back to the input's variability. Upon inspection, only two of the samples started with pearls. All others started with powder. There were no samples that started as dry aggregate or fluff.

4.3.8 CNT Synthesis

The way the CNTs were synthesized was shown to be one of the least important factors. It should be expected that the synthesis has relatively little effect on the outcome, since all the other inputs are materials, circumstances, or methods specifically tailored to increasing dispersion. However, the data shows that of the 48 data points used in the analysis, only two of them used arc discharge CNTs. The remaining 46 used HiPco CNTs. Again, this is essentially a binary variable. Moreover, it has very little spread in the data, so the domain is not well explored. From what the data can offer, the result is expected. However, to get a more meaningful result, more data is required.

4.4 Analysis of Variance

ANOVA was done to compare against our sensitivity analysis. We discussed ANOVA in Section 2.4. Due to the limited and scattered nature of our data, a full n-way interaction analysis was not possible. Furthermore, this prevented binary inputs and inputs with low variance from being included in the analysis. Therefore, in performing this analysis, we have assumed an additive model, or one with little to no interaction among inputs.

The results of the ANOVA showed that only the "Sonication Time" input rejects the null hypothesis with at least 99.9% confidence. This means variation in "Sonication Time" alone affects the mean of the output. It does not, however, suggest that other inputs can not affect the output. Variations in two or more inputs together could cause a change; this would go beyond our additive model assumption. These results confirm our most sensitive input result. However, the data is not statistically sufficient to confirm (or deny) our whole sensitivity analysis.

4.5 Modifying the Database for New Training

This section seeks to further confirm our sensitivity results. We will modify the database such that certain inputs will be removed. We will draw conclusions based on how well SFA performs with the modified data.

4.5.1 Training with Sensitive Inputs

In this test, all the input variables were eliminated except for the four (4) most sensitive ones. This is not a suggestion that these inputs are the only important factors, but that they are the only ones that we have a good distribution of data for.

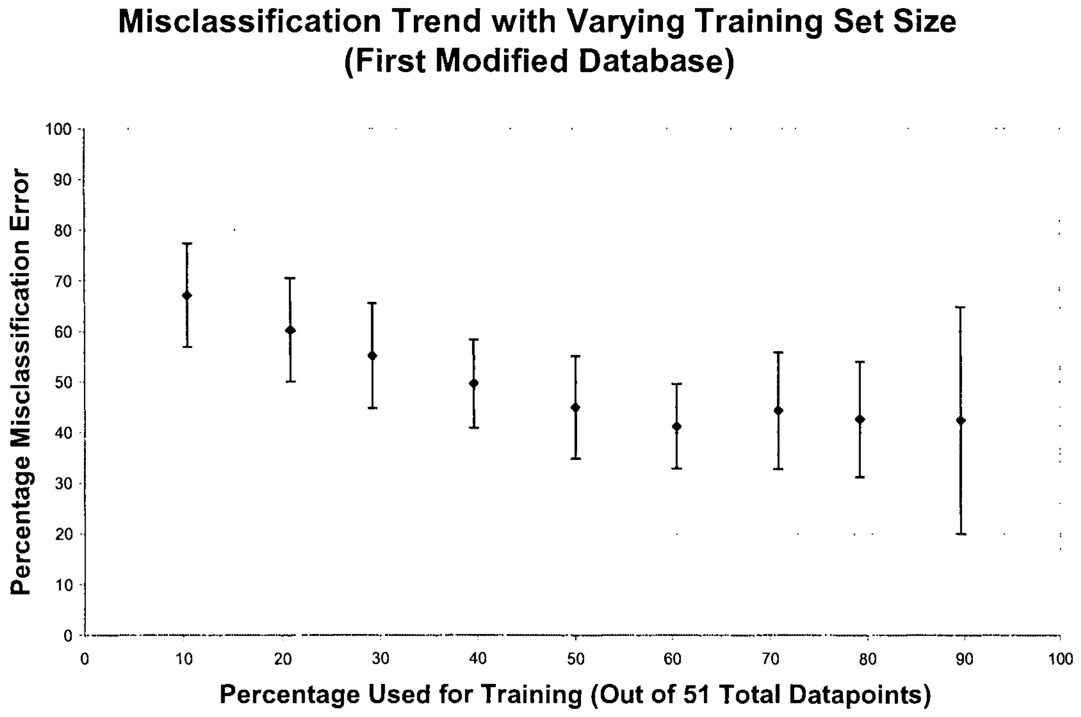

Figure 4.10. Misclassification trend with varying training set size for first modified database. All but the four most sensitive inputs were removed. Training was performed once again with varied number of training points. The downward trend, which strongly resembles that in Figure 4.1, suggests the improvement of accuracy with more training points. Due to the strong resemblance to Figure 4.1, it is reasonable to conclude that a comparable model can be achieved with just those four inputs.

Figure 4.10 shows the misclassification trend for the modified database. Again, there is a clear improvement with more training points. Also, the error bars also grow in size, as explained previously. The misclassification error closely resembles the results from the unmodified data. This suggests that SFA performs just as well with only the four most sensitive inputs. Figure 4.11 shows the input sensitivities based on 90% used for training. These sensitivities are not normalized with each other, but they are scaled by the range of their respective inputs. The results are also similar to that of the unmodified data.

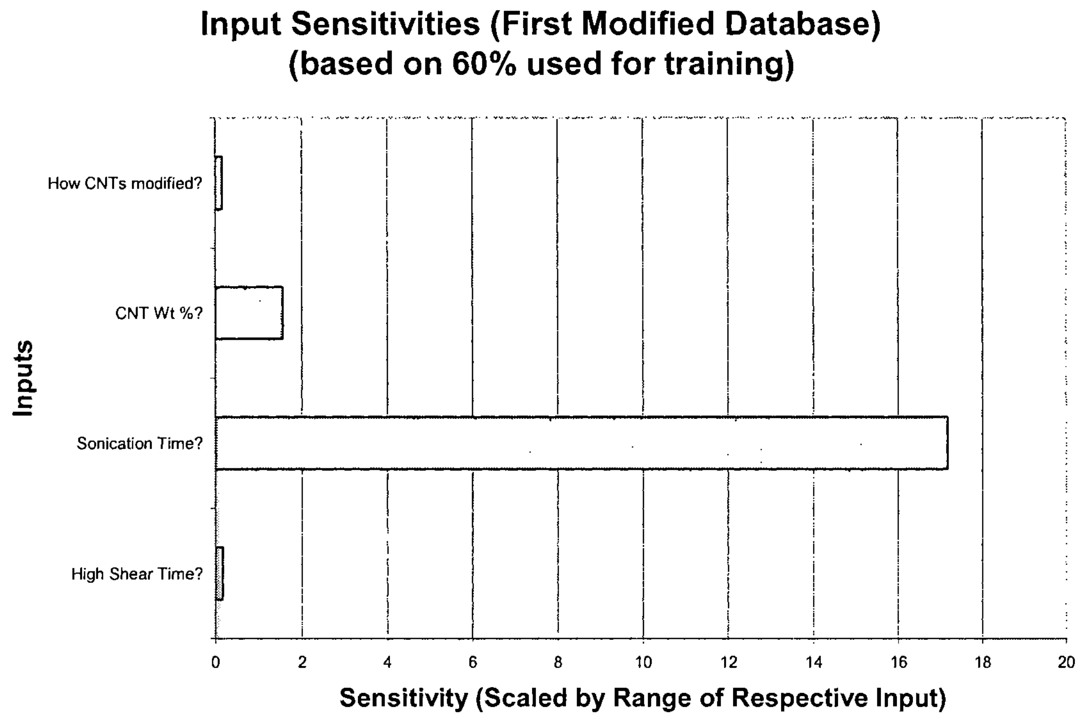

Figure 4.11. Average input sensitivities scaled by range of respective input for first modified database. Input sensitivities here were not normalized with respect to each other. They were, however, scaled by the range of the respective input, as done before. Only the four most sensitive inputs were used in these models, hence only four bars. These sensitivities were computed based on 90% of the data used for training. The sensitivities resemble Figure 4.8 (in relative magnitude), suggesting a comparable model with just these four inputs.

Figure 4.12 to Figure 4.15 plot the confusion matrix histograms for the modified database. Figure 4.12, Figure 4.13, and Figure 4.15 show good results, analogously resembling Figure 4.2, Figure 4.3, and Figure 4.5. Figure 4.14 shows worse results than the already poor results from Figure 4.4. This goes to show that there is indeed confusion (between Classes 3 and 4) which is partially cleared up by the removed inputs. Since the results are almost directly analogous to Figure 4.2 to Figure 4.5, there is further evidence that these four inputs are indeed the most sensitive.

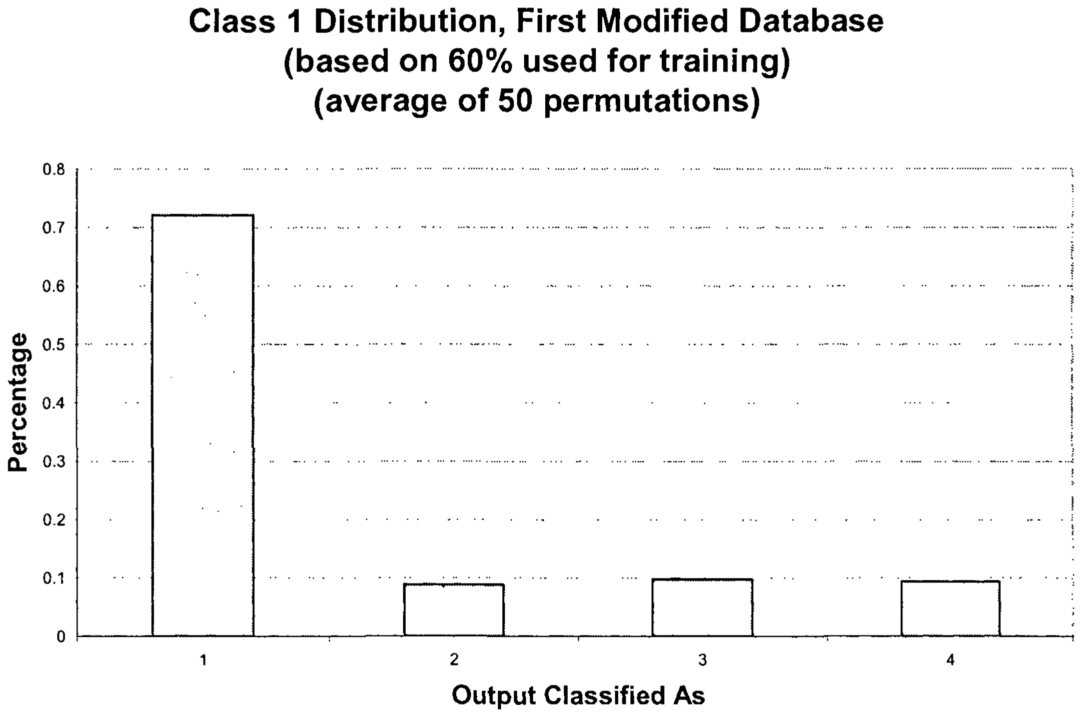

Figure 4.12. Distribution of classification for Class 1 for the first modified database.

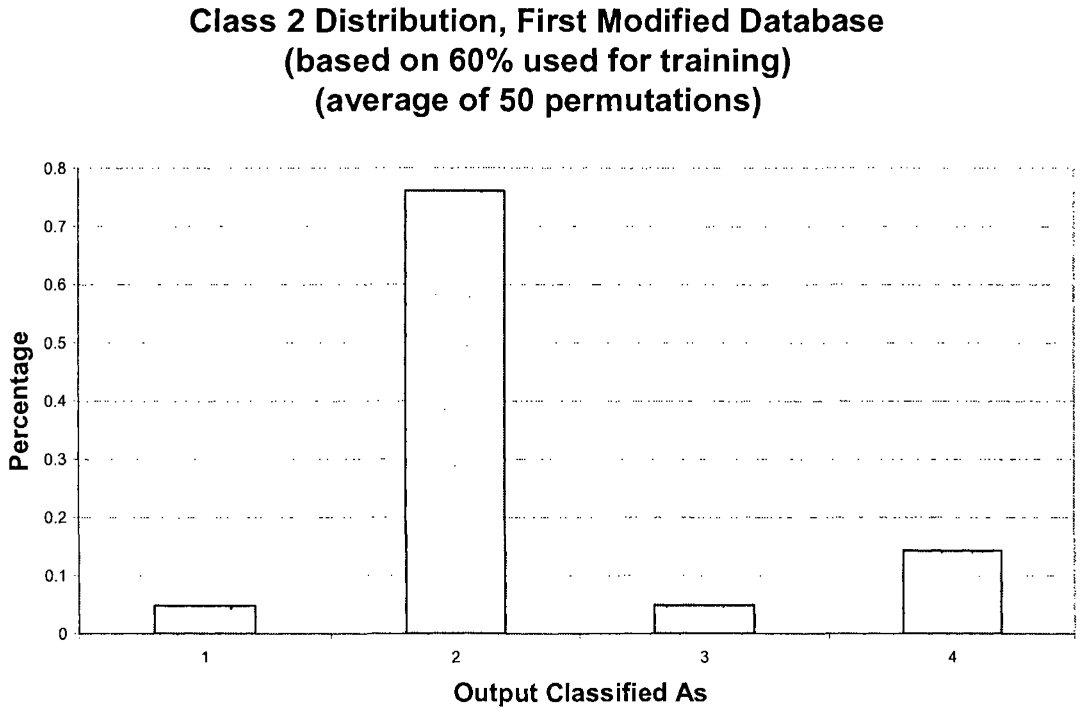

Figure 4.13. Distribution of classification for Class 2 for the first modified database.

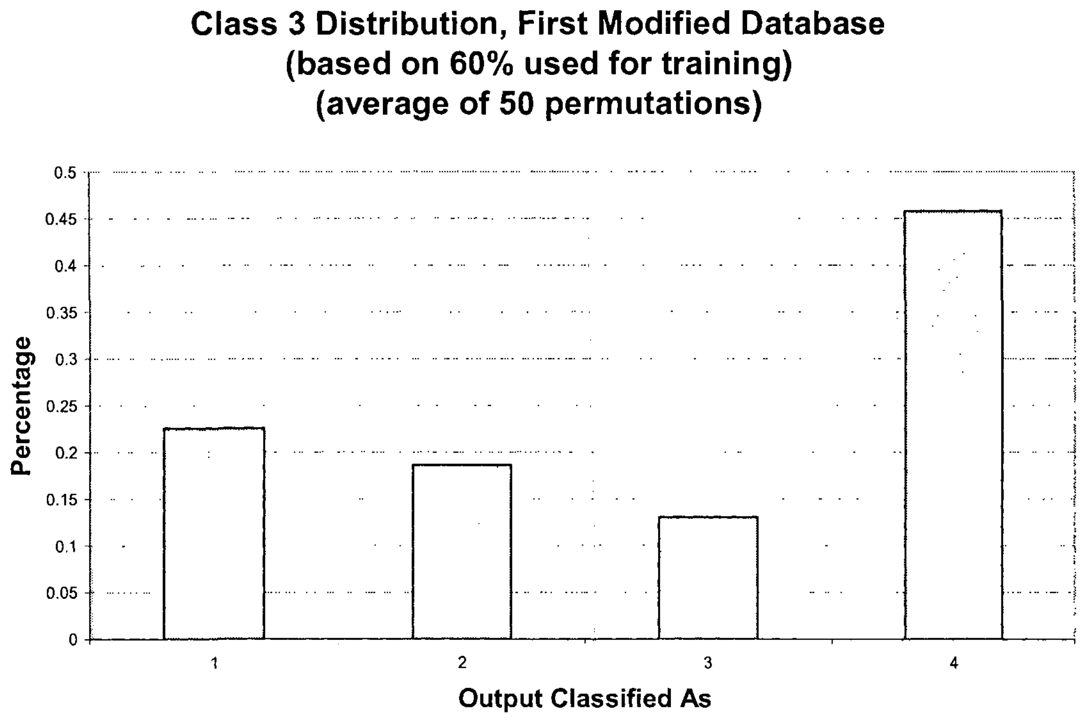

Figure 4.14. Distribution of classification for Class 3 for the first modified database.

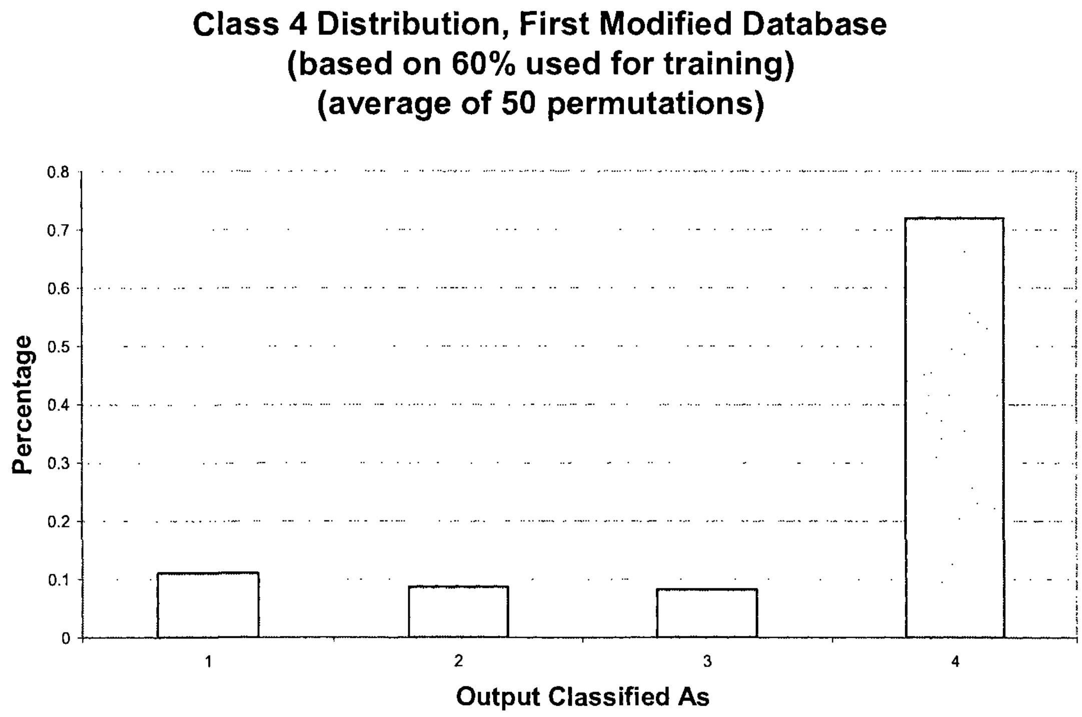

Figure 4.15. Distribution of classification for Class 4 for the first modified database.

4.5.2 Training without Sensitive Inputs

Further tests were done by training with all but the four (4) most sensitive inputs. The accuracy results are shown in Figure 4.16. One can clearly see that the percentage misclassified is very high and remains high despite increasing size of the training set. Because the models are so poor, there is no reason to show the sensitivity plot.

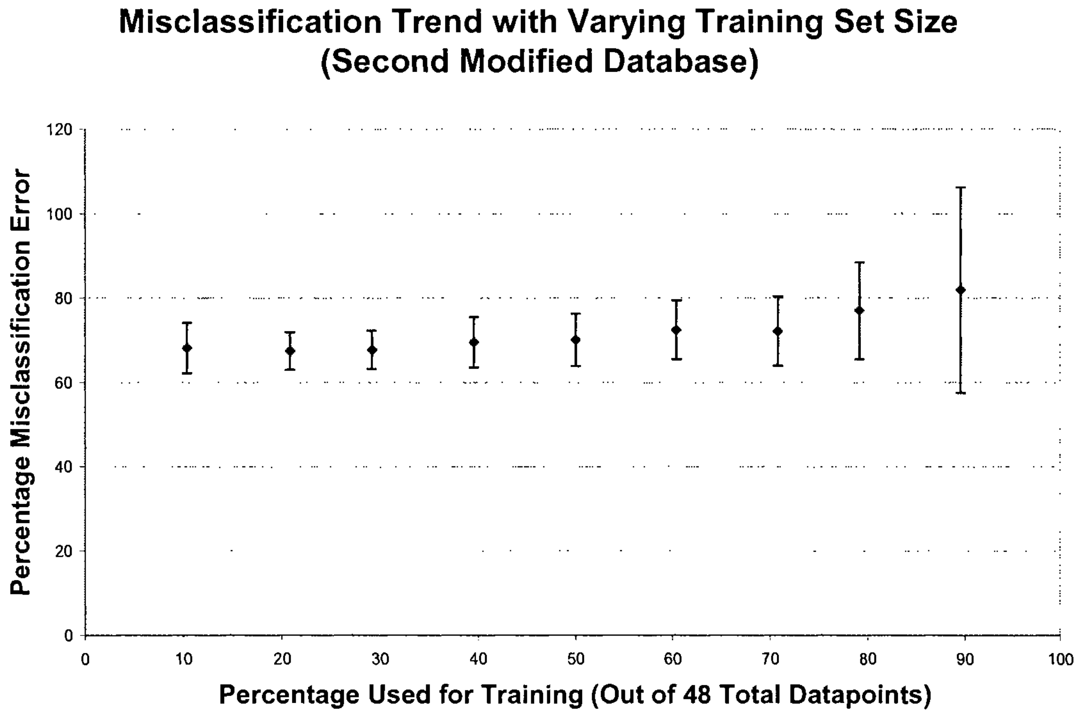

Figure 4.16. Misclassification trend with varying training set size for second modified database. Training was done with all but the four most sensitive inputs. Training and testing was performed as it was previously. Here, the misclassification error is high and remains high despite adding more training points. This suggests that a proper model can not be achieved without the inclusion of the four omitted inputs.

The results clearly do not change much when changing the original database to the first modified database. However, the results worsen drastically with the second modified database. Therefore, we can conclude that the four (4) most sensitive inputs are indeed the most important variables (given the data that we have currently acquired).

4.6 Latin Hypercube Sampling

The analysis was repeated with Latin hypercube sampling (LHS). Rather than randomly selecting training points, LHS chooses training points which are well distributed in the domain space. The range of each input variable is divided into s bins. Effectively, we have s^n sub-domains, where n is the number of input variables. Subsequently, s points are selected such that no two points should lie in a bin. The points are generated as follows:

$$x_j^{(i)} = \frac{\pi_j^{(i)} + U_j^i}{s}, \quad \forall i, j \quad \begin{cases} 1 \leq i \leq s \\ 1 \leq j \leq n, \end{cases} \tag{4.1}$$

where $U \in [0,1]$ is a uniform random number, and π is an independent random permutation of the sequence of integers 0, 1,..., $s-1$. The subscript denotes the input variable number. The superscript denotes the training point number. In practice, the training points are then selected by taking the points that lie nearest (spatially in the domain) to the generated points.

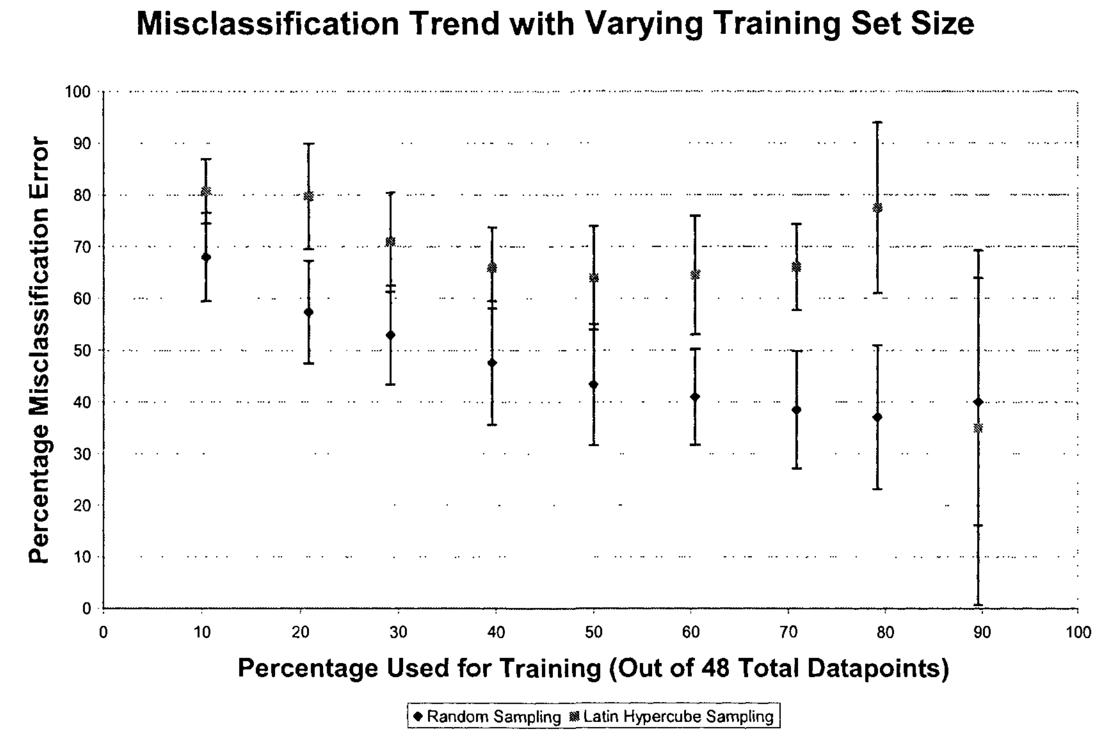

Figure 4.17. Misclassification trend with varying training set size with Latin Hypercube sampling. The trend denoted by boxes shows the LHS, compared against the random sampling denoted by diamonds. Both were performed with 50 permutations. The error bars denote standard deviations. The LHS trend is significantly worse than the random sampling. This is probably due to selecting spatially well distributed points from data which is spatially poorly distributed.

Figure 4.17 shows the misclassification trend for LHS juxtaposed with Figure 4.1 (from random sampling). Both trends plot the mean of the results after 50 permutations of the respective sampling procedure. As before, the error bars denote the standard deviations. Clearly, the LHS trend is significantly worse than that of the random sampling. This result seems counter-intuitive because LHS is designed to select points which should result in better training. However, the issue is resolved when one considers the database. Roughly half of the inputs are categorical. Of the categorical inputs, most are either binary or nearly binary. Moreover, the variability in the data is, in general, not very good; even some of the continuous variables display data which appear categorical. Also, some of the input values are not uniformly spaced. For example, "CNT Weight

Percent" takes values of 0.1, 0.5, 1, 1.5, 2, 2.5, 5, 7.5, and 10. The points are more concentrated at the lower end and more sparse at the higher end.

In essence, LHS assumes that the data has good variability in a continuous and uniformly spaced domain. Arguably at least six of our inputs have poor variability. Seven inputs are categorical. Lastly and most conclusively, two of the most influential inputs (as judged by our sensitivity analysis) are not spaced uniformly. All of these are likely reasons to cause LHS to provide a poor sampling for training data.

Chapter 5

Conclusions and Future Work

This thesis applied a computer learning algorithm to a problem in composites engineering. The struggle to find the optimal conditions for a good dispersion of CNTs contains a large number of inputs, including starting materials and details of processing procedures. Rather than test an infinite number of permutations to find those "perfect" conditions, we took what data we had and had SFA predict the rest of the domain. In this research, we organized the most common factors and processes into a series of inputs. We collected data based upon these inputs from a variety of sources. SFA trained upon portions of the data and tested itself against the remaining portions.

The results show improving accuracy trend with increasing training data. This is a logical result, and is corroborated by our simulated tests in Section 2.4. The results also revealed the four most sensitive inputs: "CNT Weight Percent," "Sonication Time," "CNT Modification," and "High Shear Mixing Time." Given our current knowledge of the domain as well as our understanding of SFA, these results are well within reason. The least sensitive inputs, "Incipient Wetting," "Shortened CNTs," "Form of Received CNTs," "CNT Synthesis," and "High Shear Mixing Rate," were also reasoned out with our physical and mathematical understandings.

For future research, at least one thing is certain. It was shown several times in this thesis that more data will provide better accuracy. In Section 2.4, we examined idealized datasets with 1000 data points. Those examples achieved extremely high accuracy. Later, in Chapter 4, we applied our real data with varying training set sizes.

With more training data, the accuracy improved consistently (except when we trained without the most sensitive inputs). A significantly larger database will not only improve accuracy, it will also explore a larger portion of the domain. Inputs which were not well explored in this research suffered and were, in some cases, deemed unimportant.

For the future, it would also be worthwhile to create an online database where any researcher can add their own data to the database. Additional inputs could be considered as well (so long as "null responses" can be conveniently and logically assigned). This would serve to increase the growth rate of the database and sample from a larger variety of sources.

References

[1] S. Iijima, "Helical microtubules of graphitic carbon," *Nature*, vol. 354, Nov. 1991, pp. 56-58.

[2] C.N.R. Rao, B.C. Satishkumar, A. Govindaraj, and M. Nath, "Nanotubes," *ChemPhysChem*, vol. 2, 2001, pp. 78-105.

[3] M.S. Dresselhaus, G. Dresselhaus, and R. Saito, "Physics of carbon nanotubes," *Carbon*, vol. 33, 1995, pp. 883-891.

[4] C. Ma, W. Zhang, Y. Zhu, L. Ji, R. Zhang, N. Koratkar, and J. Liang, "Alignment and dispersion of functionalized carbon nanotubes in polymer composites induced by an electric field," *Carbon*, vol. 46, Apr. 2008, pp. 706-710.

[5] J. Park, P. Alegaonkar, S. Jeon, and J. Yoo, "Carbon nanotube composite: Dispersion routes and field emission parameters," *Composites Science and Technology*, vol. 68, Mar. 2008, pp. 753-759.

[6] A. Funck and W. Kaminsky, "Polypropylene carbon nanotube composites by in situ polymerization," *Composites Science and Technology*, vol. 67, Apr. 2007, pp. 906-915.

[7] M.L. Shofner, "Nanotube Reinforced Thermoplastic Polymer Matrix Composites," Rice University, 2004.

[8] J.N. Coleman, U. Khan, W.J. Blau, and Y.K. Gun'ko, "Small but strong: A review of the mechanical properties of carbon nanotube-polymer composites," *Carbon*, vol. 44, Aug. 2006, pp. 1624-1652.

[9] C. Park, Z. Ounaies, K.A. Watson, R.E. Crooks, J. Smith, S.E. Lowther, J.W. Connell, E.J. Siochi, J.S. Harrison, and T.L.S. Clair, "Dispersion of single wall carbon nanotubes by in situ polymerization under sonication," *Chemical Physics Letters*, vol. 364, Oct. 2002, pp. 303-308.

[10] H. Sato and M. Sano, "Characteristics of ultrasonic dispersion of carbon nanotubes aided by antifoam," *Colloids and Surfaces A: Physicochemical and Engineering Aspects*, vol. 322, Jun. 2008, pp. 103-107.

[11] S. Pegel, P. P□tschke, G. Petzold, I. Alig, S.M. Dudkin, and D. Lellinger, "Dispersion, agglomeration, and network formation of multiwalled carbon nanotubes in polycarbonate melts," *Polymer*, vol. 49, Feb. 2008, pp. 974-984.

[12] G. Lee, S. Jagannathan, H.G. Chae, M.L. Minus, and S. Kumar, "Carbon nanotube dispersion and exfoliation in polypropylene and structure and properties of the resulting composites," *Polymer*, vol. 49, Apr. 2008, pp. 1831-1840.

[13] Y. Liao, O. Marietta-Tondin, Z. Liang, C. Zhang, and B. Wang, "Investigation of the dispersion process of SWNTs/SC-15 epoxy resin nanocomposites," *Materials Science and Engineering A*, vol. 385, Nov. 2004, pp. 175-181.

[14] C. Mitchell, J. Bahr, S. Arepalli, J. Tour, and R. Krishnamoorti, "Dispersion of Functionalized Carbon Nanotubes in Polystyrene," *Macromolecules*, vol. 35, Nov. 2002, pp. 8825-8830.

[15] L. Pena-Paras, "The role of dispersion of carbon nanotubes on the physical properties of polymer nanocomposites," Rice University.

[16] A. Srivastava, A. Meade, and J. Needham, "Adaptive Algorithm for Classifying Launch and Recovery of the HH-60H Seahawk," *Journal of Aerospace Computing, Information, and Communication*, vol. 6, 2009, pp. 109-120.

[17] A. Keane and P. Nair, *Computational Approaches for Aerospace Design: The Pursuit of Excellence*, Wiley, 2005.

[18] T. Hastie, R. Tibshirani, and J.H. Friedman, *The Elements of Statistical Learning*, Springer, 2003.

[19] R.A. DeVore, "Nonlinear Approximation," *Acta Numerica*, 1998, pp. 51-150.

[20] E. Schmidt, "Zur Theorie der linearen und nichtlinearen Integralgleichungen," *Mathematische Annalen*, vol. 63, Dec. 1907, pp. 433-476.

[21] V.M. Temlyakov, "Weak Greedy Algorithms," *IMI Research Report*, vol. 99, University of South Carolina. 1999.

[22] L. Jones, "Constructive approximations for neural networks by sigmoidal functions," *Proceedings of the IEEE*, vol. 78, 1990, pp. 1586-1589.

[23] L.K. Jones, "A Simple Lemma on Greedy Approximation in Hilbert Space and Convergence Rates for Projection Pursuit Regression and Neural Network Training," *The Annals of Statistics*, vol. 20, Mar. 1992, pp. 608-613.

[24] A. Barron, "Universal approximation bounds for superpositions of a sigmoidal function," *Information Theory, IEEE Transactions on*, vol. 39, 1993, pp. 930-945.

[25] C.A.J. Fletcher, *Computational Galerkin Methods: Springer Series in Computational Physics*, Springer-Verlag Telos, 1984.

[26] A. Beygelzimer, J. Langford, and B. Zadrozny, "Weighted One-Against-All," *AAAI*, AAAI Press / The MIT Press, 2005, pp. 720-725.

[27] A. Berger, "Error-correcting output coding for text classification," *In Proceedings of IJCAI-99 Workshop on Machine Learning for Information Filtering*, 1999.

[28] F. Masulli and G. Valentini, "Comparing Decomposition Methods for Classication."

[29] R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification," *J. Mach. Learn. Res.*, vol. 5, 2004, pp. 101-141.

[30] R. Collobert, S. Bengio, and C. Williamson, "SVMTorch: Support vector machines for large-scale regression problems," *Journal of Machine Learning Research*, vol. 1, 2001, pp. 143--160.

[31] M. Seo and S. Park, "Electrical resistivity and rheological behaviors of carbon nanotubes-filled polypropylene composites," *Chemical Physics Letters*, vol. 395, Sep. 2004, pp. 44-48.

[32] K.H. Kim and W.H. Jo, "Improvement of tensile properties of poly(methyl methacrylate) by dispersing multi-walled carbon nanotubes functionalized with poly(3-hexylthiophene)-graft-poly(methyl methacrylate)," *Composites Science and Technology*, vol. 68, Jul. 2008, pp. 2120-2124.

[33] Q. Li, Q. Xue, L. Hao, X. Gao, and Q. Zheng, "Large dielectric constant of the chemically functionalized carbon nanotube/polymer composites," *Composites Science and Technology*, vol. 68, Aug. 2008, pp. 2290-2296.

[34] S. Bao and S. Tjong, "Mechanical behaviors of polypropylene/carbon nanotube nanocomposites: The effects of loading rate and temperature," *Materials Science and Engineering: A*, vol. 485, Jun. 2008, pp. 508-516.

[35] G. Liang, S. Bao, and S. Tjong, "Microstructure and properties of polypropylene composites filled with silver and carbon nanotube nanoparticles prepared by melt-compounding," *Materials Science and Engineering: B*, vol. 142, Sep. 2007, pp. 55-61.

[36] T. Kashiwagi, E. Grulke, J. Hilding, K. Groth, R. Harris, K. Butler, J. Shields, S. Kharchenko, and J. Douglas, "Thermal and flammability properties of polypropylene/carbon nanotube nanocomposites," *Polymer*, vol. 45, May. 2004, pp. 4227-4239.

[37] P. Potschke, B. Kretzschmar, and A. Janke, "Use of carbon nanotube filled polycarbonate in blends with montmorillonite filled polypropylene," *Composites Science and Technology*, vol. 67, Apr. 2007, pp. 855-860.

[38] A. Loiseau, P. Launois, P. Petit, S. Roche, and J. Salvetat, *Understanding Carbon Nanotubes: From Basics to Applications*, Springer, 2006.

[39] T. Guo, P. Nikolaev, A.G. Rinzler, D. Tomanek, D.T. Colbert, and R.E. Smalley, "Self-Assembly of Tubular Fullerenes," *Journal of Physical Chemistry*, vol. 99, Jul. 1995, pp. 10694-10697.

[40] M. Jose-Yacaman, M. Miki-Yoshida, L. Rendon, and J.G. Santiesteban, "Catalytic growth of carbon microtubules with fullerene structure," *Applied Physics Letters*, vol. 62, Feb. 1993, pp. 657-659.

[41] W.F. Smith, *Principles of Materials Science and Engineering : Third Edition*, Mc Graw-Hill Book Company, 1999.

[42] I. Chiang, B. Brinson, A. Huang, P. Willis, M. Bronikowski, J. Margrave, R. Smalley, and R. Hauge, "Purification and Characterization of Single-Wall Carbon Nanotubes (SWNTs) Obtained from the Gas-Phase Decomposition of CO (HiPco Process)," *Journal of Physical Chemistry B*, vol. 105, Sep. 2001, pp. 8297-8301.

[43] K.B. Shelimov, R.O. Esenaliev, A.G. Rinzler, C.B. Huffman, and R.E. Smalley, "Purification of single-wall carbon nanotubes by ultrasonically assisted filtration," *Chemical Physics Letters*, vol. 282, Jan. 1998, pp. 429-434.

[44] V. Khabashesku, J. Margrave, and E. Barrera, "Functionalized carbon nanotubes and nanodiamonds for engineering and biomedical applications," *Diamond and Related Materials*, vol. 14, 2005, pp. 859-866.

[45] J. Chen, M.A. Hamon, H. Hu, Y. Chen, A.M. Rao, P.C. Eklund, and R.C. Haddon, "Solution Properties of Single-Walled Carbon Nanotubes," *Science*, vol. 282, Oct. 1998, pp. 95-98.

[46] X. Jiang, Y. Bin, and M. Matsuo, "Electrical and mechanical properties of polyimide-carbon nanotubes composites fabricated by in situ polymerization," *Polymer*, vol. 46, Aug. 2005, pp. 7418-7424.

[47] E.S. Richardson, W.G. Pitt, and D.J. Woodbury, "The Role of Cavitation in Liposome Formation," *Biophys. J.*, vol. 93, Dec. 2007, pp. 4100-4107.

[48] M. Lopez Manchado, L. Valentini, J. Biagiotti, and J. Kenny, "Thermal and mechanical properties of single-walled carbon nanotubes-polypropylene composites prepared by melt processing," *Carbon*, vol. 43, Jun. 2005, pp. 1499-1505.

[49] M. Baibarac, I. Baltog, S. Lefrant, J. Mevellec, and C. Bucur, "Vibrational and photoluminescence properties of the polystyrene functionalized single-walled carbon nanotubes," *Diamond and Related Materials*, vol. In Press, Corrected Proof.

[50] A.R. Bhattacharyya, T.V. Sreekumar, T. Liu, S. Kumar, L.M. Ericson, R.H. Hauge, and R.E. Smalley, "Crystallization and orientation studies in polypropylene/single wall carbon nanotube composite," *Polymer*, vol. 44, Apr. 2003, pp. 2373-2377.

[51] E. Camponeschi, R. Vance, M. Al-Haik, H. Garmestani, and R. Tannenbaum, "Properties of carbon nanotube-polymer composites aligned in a magnetic field," *Carbon*, vol. 45, Sep. 2007, pp. 2037-2046.

[52] T. Chang, L. Jensen, A. Kisliuk, R. Pipes, R. Pyrz, and A. Sokolov, "Microscopic mechanism of reinforcement in single-wall carbon nanotube/polypropylene nanocomposite," *Polymer*, vol. 46, Jan. 2005, pp. 439-444.

[53] T. Chang, A. Kisliuk, S. Rhodes, W. Brittain, and A. Sokolov, "Conductivity and mechanical properties of well-dispersed single-wall carbon nanotube/polystyrene composite," *Polymer*, vol. 47, Oct. 2006, pp. 7740-7746.

[54] J. Dai, Q. Wang, W. Li, Z. Wei, and G. Xu, "Properties of well aligned SWNT modified poly (methyl methacrylate) nanocomposites," *Materials Letters*, vol. 61, Jan. 2007, pp. 27-29.

[55] D.O. McIntosh, "Mechanical Properties of Fibers Made from Single-walled Carbon Nanotube Reinforced Thermoplastic Polymer Composites," Rice University, 2005.

[56] Z. Ounaies, C. Park, K.E. Wise, E.J. Siochi, and J.S. Harrison, "Electrical properties of single wall carbon nanotube reinforced polyimide composites," *Composites Science and Technology*, vol. 63, Aug. 2003, pp. 1637-1646.

[57] F.J. Owens, "Properties of composites of fluorinated single walled carbon nanotubes and polyacrylonitrile," *Materials Letters*, vol. 59, Dec. 2005, pp. 3720-3723.

[58] M. Pulikkathara, "Unpublished results," Rice University, 2008.

[59] B. Sitharaman, X. Shi, L.A. Tran, P.P. Spicer, I. Rusakova, L.J. Wilson, and A.G. Mikos, "Injectable in situ cross-linkable nanocomposites of biodegradable polymers

and carbon nanostructures for bone tissue engineering," *Journal of Biomaterials Science, Polymer Edition*, vol. 18, Jun. 2007, pp. 655-671.

[60] L. Sun, G. Warren, J. O'Reilly, W. Everett, S. Lee, D. Davis, D. Lagoudas, and H. Sue, "Mechanical properties of surface-functionalized SWCNT/epoxy composites," *Carbon*, vol. 46, Feb. 2008, pp. 320-328.

[61] L. Valentini, J. Biagiotti, J.M. Kenny, and S. Santucci, "Morphological characterization of single-walled carbon nanotubes-PP composites," *Composites Science and Technology*, vol. 63, Jun. 2003, pp. 1149-1153.

[62] L. Valentini, D. Puglia, F. Carniato, E. Boccaleri, L. Marchese, and J.M. Kenny, "Use of plasma fluorinated single-walled carbon nanotubes for the preparation of nanocomposites with epoxy matrix," *Composites Science and Technology*, vol. 68, Mar. 2008, pp. 1008-1014.

[63] Q. Zhang, S. Rastogi, D. Chen, D. Lippits, and P.J. Lemstra, "Low percolation threshold in single-walled carbon nanotube/high density polyethylene composites prepared by melt processing technique," *Carbon*, vol. 44, Apr. 2006, pp. 778-785.

[64] J. Zhu, J. Kim, H. Peng, J. Margrave, V. Khabashesku, and E. Barrera, "Improving the Dispersion and Integration of Single-Walled Carbon Nanotubes in Epoxy Composites through Functionalization," *Nano Letters*, vol. 3, Aug. 2003, pp. 1107-1113.

CNTs made?	CNTs received?	CNTs modified?	Shortened?	Solvent?	Weight percent?	Polymer?	Incipient wetting?	Magnetic mixing time?	Homogenization time?	Sonication time?	High shear start temp?	High shear stop temp?	High shear time?	High shear rate?	Dispersion output
1	2	1	0	2	5	0	1	0	0	40	165	165	12	75	1
1	2	1	0	2	10	0	1	0	0	40	165	165	12	75	2
1	2	7	0	2	2.5	0	1	0	0	40	40	175	12	75	3
1	2	7	0	2	10	0	1	0	0	40	40	175	12	75	3
1	2	1	0	4	1.5	2	1	0	0	30	125	125	12	60	1
1	2	2	0	3	1.5	2	1	0	0	30	125	125	12	60	4
1	2	1	1	2	1	2	1	0	0	60	115	115	12	75	1
1	2	2	1	2	1	2	1	0	0	60	115	115	12	75	3
1	2	1	0	6	1	9	0	0	0	180	30	30	0	0	3
1	2	11	0	6	1	9	0	0	0	180	30	30	0	0	4
3	2	2	0	10	0.1	13	0	0	0	30	30	30	0	0	4
3	2	0	0	10	0.1	13	0	0	0	30	30	30	0	0	2
1	1	1	0	8	1	9	0	30	5	0	30	30	0	0	1
1	1	12	0	8	1	9	0	30	5	0	30	30	0	0	4
1	2	0	0	2	0.5	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	1	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	2	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	2.5	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	5	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	7.5	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	10	0	1	0	0	60	165	165	10	75	2
1	2	0	0	2	15	0	1	0	0	60	165	165	10	75	2
1	2	7	0	2	0.5	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	1	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	2	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	2.5	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	5	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	7.5	0	1	0	0	60	165	165	15	75	4
1	2	7	0	2	10	0	1	0	0	60	165	165	15	75	4
1	2	1	0	0	0.5	5	0	0	0	1	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	3	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	4	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	5	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	8	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	11	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	16	30	30	0	0	1
1	2	1	0	0	0.5	5	0	0	0	18	30	30	0	0	3
1	2	1	0	0	0.5	5	0	0	0	21	30	30	0	0	3
1	2	1	0	0	0.5	5	0	0	0	33	30	30	0	0	3
1	2	1	0	0	0.5	5	0	0	0	37	30	30	0	0	3
1	2	1	0	0	0.5	5	0	0	0	42	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	55	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	65	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	85	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	98	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	106	30	30	0	0	4
1	2	1	0	0	0.5	5	0	0	0	127	30	30	0	0	3
1	2	1	0	0	0.5	5	0	0	0	254	30	30	0	0	3

Table A.1. Database of 48 Entries

CNTs modified?	Weight percent?	Sonication time?	High shear time?	Dispersion output
1	5	40	12	1
1	10	40	12	2
7	2.5	40	12	3
7	10	40	12	3
1	1.5	30	12	1
2	1.5	30	12	4
1	1	60	12	1
2	1	60	12	3
0	0.1	120	10	4
0	1	120	10	4
1	0.5	360	0	4
1	1	180	0	3
11	1	180	0	4
2	0.1	30	0	4
0	0.1	30	0	2
1	1	0	0	1
12	1	0	0	4
0	0.5	60	10	2
0	1	60	10	2
0	2	60	10	2
0	2.5	60	10	2
0	5	60	10	2
0	7.5	60	10	2
0	10	60	10	2
0	15	60	10	2
7	0.5	60	15	4
7	1	60	15	4
7	2	60	15	4
7	2.5	60	15	4
7	5	60	15	4
7	7.5	60	15	4
7	10	60	15	4
1	0.5	1	0	1
1	0.5	3	0	1
1	0.5	4	0	1
1	0.5	5	0	1
1	0.5	8	0	1
1	0.5	11	0	1
1	0.5	16	0	1
1	0.5	18	0	3
1	0.5	21	0	3
1	0.5	33	0	3
1	0.5	37	0	3
1	0.5	42	0	4
1	0.5	55	0	4
1	0.5	65	0	4
1	0.5	85	0	4
1	0.5	98	0	4
1	0.5	106	0	4
1	0.5	127	0	3
1	0.5	254	0	3

Table A.2. First Modified Database of 51 Entries

CNTs made?	CNTs received?	Shortened?	Solvent?	Polymer?	Incipient wetting?	Magnetic mixing time?	Homogenization time?	High shear start temp?	High shear stop temp?	High shear rate?	Dispersion output
1	2	0	2	0	1	0	0	165	165	75	1
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	40	175	75	3
1	2	0	2	0	1	0	0	40	175	75	3
1	2	0	4	2	1	0	0	125	125	60	1
1	2	0	3	2	1	0	0	125	125	60	4
1	2	1	2	2	1	0	0	115	115	75	1
1	2	1	2	2	1	0	0	115	115	75	3
1	2	0	6	9	0	0	0	30	30	0	3
1	2	0	6	9	0	0	0	30	30	0	4
3	2	0	10	13	0	0	0	30	30	0	4
3	2	0	10	13	0	0	0	30	30	0	2
1	1	0	8	9	0	30	5	30	30	0	1
1	1	0	8	9	0	30	5	30	30	0	4
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	2
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	2	0	1	0	0	165	165	75	4
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	1
1	2	0	0	5	0	0	0	30	30	0	3
1	2	0	0	5	0	0	0	30	30	0	3
1	2	0	0	5	0	0	0	30	30	0	3
1	2	0	0	5	0	0	0	30	30	0	3
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	4
1	2	0	0	5	0	0	0	30	30	0	3
1	2	0	0	5	0	0	0	30	30	0	3

Table A.3. Second Modified Database of 48 Entries.