Exploratory Data Analysis for Preemptive Quality Control

by

Kaan Karamancı

B.S., Massachusetts Institute of Technology (2008)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author ....[illegible]..........
Department of Electrical Engineering and Computer Science
May 22, 2009

Certified by... [illegible] ....
Stanley B. Gershwin
Senior Research Scientist
Thesis Supervisor

Accepted by ..[illegible]..........
Arthur C. Smith
Chairman, Department Committee on Graduate Theses

Exploratory Data Analysis for Preemptive Quality Control

by

Kaan Karamancı

Submitted to the Department of Electrical Engineering and Computer Science
on May 22, 2009, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

In this thesis, I proposed and implemented a methodology to perform preemptive quality control on low-tech industrial processes with abundant process data. This involves a 4 stage process which includes understanding the process, interpreting and linking the available process parameter and quality control data, developing an exploratory data toolset and presenting the findings in a visual and easily implementable fashion. In particular, the exploratory data techniques used rely on visual human pattern recognition through data projection and machine learning techniques for clustering. The presentation of finding is achieved via software that visualizes high dimensional data with Chernoff faces. Performance is tested on both simulated and real industry data. The data obtained from a company was not suitable, but suggestions on how to collect suitable data was given.

Thesis Supervisor: Stanley B. Gershwin
Title: Senior Research Scientist

Acknowledgments

I believe who we are and what we become is a function of the people who have touched our lives through the years. In these few paragraphs, I would like to thank several people without whom this thesis, my academic career and, indeed, my life as I knew it would not have taken shape.

First, I would like to thank my supervisors Dr. Stanley Gerschwin and Dr. Irvin Schick for their continuous support and patient guidance without which this thesis would not have been possible. Always kind and patient with my struggles, trials and errors, they are the best advisors that one could hope to work with. Thank you.

I would also like to thank my academic advisor, Dr. Jeffrey Shapiro who guided me throughout my tenure at MIT and Dr. Patrick Winston for imparting on me the greatest academic and life experiences I have attained to date.

I am grateful to Nuri Özgür. His guidance through my high school years made it possible for me to pursue higher education at MIT. I would also like to thank my high school (Robert College of Istanbul), my teachers and friends whose endless care and support allowed me to build the foundations of my personality and knowledge. They armed me well to tackle the many trials of MIT. Many thanks.

To my closest friends Fırat Ileri, Rahul Shroff, Volkan Gürel, Jennifer Tang, Orhan Dağlı, Cankut Durgun, and Burkay Gür: Though the winds of life may carry us far away, we shared something special on these few blocks along the Charles. I hope that you will remember it as I do. The icy winter cold, walking towards the impending doom that is the finals is now a warm memory as I recall the silent company in which we shared. All the times that made life at MIT worth living and laughters meaningful was with you. All those times that, through my troubles, I managed a hint of a smile was thanks to you. You will be dearly missed.

Finally, I am eternally thankful to my family for their unconditional love and support. To my mother, father and sister: For the countless, sleepless nights through which you raised this child, I am forever grateful. You are the reason I'm here today. You are the reason I am. Truly, 'you are all my reasons'. This thesis, along with all my heart, is dedicated to you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Purpose

The purpose of this thesis is to present the theoretical background and methodology for the linking of process data with output data via exploratory analysis in order to detect defective product early. The process that will be used is the indigo dyeing range of a denim production factory. The methodology is implemented on simulated data verifying its validity. These methods are then applied in industry and the findings are presented. Our results from looking at Company X data showed clearly that the data did not allow for the proper analysis of inputs and outputs. We explored various reasons for this, which are summarized in the conclusion. Suggestions as to how to obtain best results are given.

1.2 Statement of Problem

Current quality control techniques for manufacturing often utilize a strategy known as PDCA (Plan-Do-Check-Act) or the Deming cycle [2]. This strategy may be considered *a posteriori* quality control because the manufacturing process is designed and carried out before the quality of the product is checked. If the product fails to perform within specifications (determined by a variety of statistical methods) this triggers action on the part of the manufacturer to look into the process and figure out the problem. While extensive literature exists on this technique and the associated statistical methods are applied in many industries with great effect (e.g. Motorola's six-sigma strategy), they do not prevent certain inefficiencies.

Specifically, the PDCA technique relies on checking products as they come out of the assembly line. Thus, if a systemic error has occurred, this error will be detected only after a number of faulty products have accumulated to reveal the error. This accumulation of faulty product leads to inefficiency of the assembly line and wastes valuable time and resources.

The problem of wasted resources becomes particularly pronounced in industry settings that incorporate one or more of the following:

High Production Lead Time: The longer a faulty product stays in development the more time/resources it wastes.

Multiple Stages of Production: The later the product is checked for quality in multi-stage production, the more unnecessary processing it goes through.

Single or Few Lines of Production: Faulty products take up valuable resources and capacity.

The primary underlying motivator of PDCA quality control, particularly as it applies to low-tech production companies, is the lack of data recording and in-depth statistical analysis of the production process. While process controllers are experts at keeping inputs within specifications; numerical methods that span multiple parameters and define the process mathematically often do not exist in low-tech industries. This leads to the requirement of careful a posteriori quality control.

An example of this, which will be explored in depth throughout this thesis, is the dyeing process in a denim production mill. The engineers that control the process are experts in chemistry and the indigo color. However, a multivariate analysis on how temperature, acidity, dye flow etc. collectively affect the color has never been performed. Therefore, while each parameter is kept in a certain range (often determined by the experience of the engineer rather than a prescription) the understanding of effects on color are limited to variations in single variables. This forces the mill to implement precise quality control at various stages in production, checking the color for accuracy and discarding fabric in large quantities that do not meet specifications.

We may state the problem of quality control in low-tech industries as the following:

- PDCA quality control accounts for a large amount of lost production and wasted machine time
- PDCA quality control is necessitated due to the lack of well-defined numerical characterization in the production process.

The goal of this thesis is to explore data-driven techniques which may be utilized in low-tech industries to limit the requirement for a posteriori checks through enhanced in-process quality control.

1.3 Objective

If we are to prevent or preemptively identify faulty end product, we must be able to identify complex patterns in the production process, taking into account multiple variations in production parameters. To do this, we must first obtain informative data that link process parameters to desirable and undesirable output values. Second we must employ both linear and nonlinear, multivariate statistical techniques to identify patterns in the data. Finally, we must interpret the results to identify sound decision strategies for process controllers. Below is a summary discussion of what may be expected - and is carried out throughout the thesis- of the three components to accomplish our objective.

1.3.1 Data

Obtaining appropriate data in low-tech settings is often more challenging than it appears. Especially in low-tech settings, in which processes are not well-defined numerically, data recording is often done for purposes other than research. The data that satisfies our needs would need to have the following properties:

Links process measurements to output results: Required to establish correlation between inputs and outputs.

Matching input, output granularity Ensures that there is a one-to-one assigment between process measurements and output measurements.

Unique sets of input and output data: Eliminates redundancy or the possibility of ambiguous results.

It is desirable that the above properties be satisfied in order to produce meaningful results from the analysis. As this thesis will show, it is sometimes not possible to obtain such data in real settings due to a number of physical and technical factors.

1.3.2 Exploratory Data Analysis

Process controllers are often aware of linear or single variable effects of process parameters on output results. However, it may also be beneficial to look at multivariate effects. This thesis focuses on this idea to identify numerical results that were previously unknown in such processes. Therefore the focus of the analysis is on the following techniques:

The Common Methods: These incorporate the most common of statistical methods (i.e. regression, principal component analysis)

Multivariate Linear Projection: This technique uses optimal linear projections to derive the most meaning from available data.

Recursive Software Optimization: This exploration is more in the spirit of chance encounters with better than random results. It implements software that *tries* projections onto lower dimension spaces for optimal separation of the data into meaningful clusters.

Support Vector Machines: SVM is used primarily in machine learning to classify various data sets. The power of the technique is that it can separate data in various non-linear forms.

These techniques allow for both a linear and non-linear exploration of the data. It is important to note that the primary objective for the exploratory data analysis is *identifying clusters* or separations in the data that may account for *within spec* vs. *out of spec* output.

1.3.3 Decision Strategies

Finally, the result of the analysis must be presented in a format that allows for controllers to utilize. The strategies put forth must therefore be mathematically sound, yet easily employed. While this thesis does not suggest a decision strategy for the industrial example explored, a generic form for a *decision boundary* as well as a supporting visual application is presented. The decision strategy objective is twofold:

Decision Boundary: A mathematical basis for in-process quality control setting multivariate limits to the process.

Visual Presentation: The user-friendly presentation of the mathematical basis so that it may be easily applied by process controllers.

The objectives of this thesis are:

- Decide upon and test applicability of exploratory data analysis techniques on the given problem.
- Employ these techniques on available data in a setting representative of low-tech industries.

- Derive a strategy to establish in-process quality control.
- Implement a visual application to facilitate preemptive quality control.

This thesis will be successful if it can provide and test the applicability of a set of methodologies in identifying patterns that may enable in-process quality control and compare the effectiveness of said methodologies for different data sets and industries.

1.4 Terminology

This thesis incorporates terminology as used in the research. Referrals to the following terminology will be made throughout the thesis:

Input, Output: These refer to the process measurements and the output values of the end product respectively

Go, No-Go: The data is explored and separated according to whether it produces a within spec (go) or out of spec (no-go) end product

Chapter 2

Background

2.1 Literature Review

The technical work of this thesis consists of two primary goals: Exploratory data analysis (EDA) and developing decision strategies. There is extensive literature in EDA methods as well as both the mathematical basis of and visual applications for decision making.

EDA is a set of statistical analysis methods that are applied to find cause and effect patterns in data. In many ways this is similar to hypothesis testing but involves more complicated techniques to reduce dimensionality or identify multivariate patterns. The primary task is to reduce the dimensionality of the data while simultaneously grouping it in order to derive meaningful patterns from 2D or 3D plots. One of the often used techniques for reducing dimensionality is principal component analysis (PCA) [6]. PCA is a linear transformation of the data points. This method uses eigenvalue decomposition of the multidimensional data and applies a linear orthogonal transformation to capture the lower order components. If the original data consists of p dimensions, PCA reduces this to $q <= p$ dimensions, where each dimension in q is a linear combination of each of the p dimensions and orthonormal to other dimensions in q. Reduction from p to q dimensions is done along the vectors of maximum variance in the data. As such, it captures the most amount of information. While this is desirable in most applications, PCA reduces data that is initially homogeneous (i.e.

belong to a single cluster or group). This differs from our goals because we are primarily concerned in grouping data into clusters which we may then identify as Go and No-Go.

However, PCA is in line with our task of capturing information in data. There is extensive literature on various forms of information and methods to find it. Out of this literature, the most general form of discovering such patterns is projection pursuit techniques. Projection pursuit is a group of techniques that linearly project high-dimensional data onto lower dimensions. These techniques differ in the types of projections, or goals of projections used. PCA and regression are two special cases of projection pursuit.

The most important advantages of projection pursuit are its ability to avoid the curse of dimensionality, provide multivariate analysis and ignore irrelevant dimensions and noise [4]. In these ways, we may view projection pursuit as a method to increase the likelihood of finding interesting patterns. An example of this would be to project high-dimensional data along axes that best capture a deviation from normal distribution. If we accept normal distribution as the least interesting distribution (as random noise is distributed in this manner), then a deviation from this distribution would be a benchmark for finding interesting distributions. The chi-square test for normality would then provide us with a valid metric to test for the best projection axes.

Within the literature on information retrieval from data, the following techniques are most often utilized: PCA, regression and chi-squared deviation. In this thesis, we will explore a unique linear projection technique developed specifically to separate the data into two clusters. This, for our purposes, is the information of our data. We will further explore software that utilizes projection pursuit to find the best separation of two clusters.

The literature thus far reduces dimensionality via linear and nonlinear projections. However, as stated, these projections serve to increase the likelihood of finding interesting patterns -or identifiable clusters in our case. As such, further techniques are necessary to find the best clusters and boundaries.

Clustering techniques belong in two groups: Hierarchical clustering and partitional clustering. Hierarchical clustering is based on the premise that two clusters already exist and new members must be classified, further adjusting the cluster and its boundaries. Therefore, it is progressive and continuously adjusting. While the progressive nature may be appealing to us to refine clusters, we are also required to define the clusters given our current data. As such, the more appealing clustering methods fall under partitional clustering, which are techniques that fit current data into two clusters. Both groups are further separated by the metrics they use.

One example of a progressive clustering algorithm would be K-means clustering. This algorithm assigns points into K clusters according to their Euclidian distance from the mean of the clusters [3]. The algorithm then recalculates and adjusts the means of the clusters. The initialization occurs with the generation of two random clusters that later conform to the data as new points are generated. The disadvantage of this is that the end result is greatly dependant on the initial clusters and the order at which points are introduced.

Most clustering techniques are derivatives of k-means, or work similarly in introducing new data. In this way, most clustering techniques do not fit well with our purposes, as we would prefer a method that looks at extensive data as a whole, rather than introduce new data one by one. Indeed, dynamic adaptability is not a primary concern for our objective, as this would only serve to refine our result because low-tech industrial processes are unlikely to change spontaneously.

Support vector machines are an effective and flexible classification tool often used in machine learning. It is particularly in line with the objectives of this thesis as it is primarily concerned with the separation of two classes of data. SVM finds the best separation hyperplane from the two classes of data, where best means the maximum distance of separation between the closest neighboring data points of the two classes [5].

The power of SVM arises from solely depending on the dot product of points to be classified. As such, it allows for the use of kernel methods: the mapping of the original data set onto higher or transformed dimensions. This allows SVM to capture patterns in linear, quadratic, cubic, radial and many more types of functions.

2.2 Company and Industry Background

As previously stated, this thesis aims to improve quality control via statistical analysis of low-tech industries. The definition of low-tech in this case does not imply that the production technology is primitive, but rather that the technology lacks, or does not require numerical and mathematical precision in process control. As an example, we may compare textile mills to microchip production. While spinning, weaving and dyeing technologies are quite advanced with precision machines, the control process for these machines does not require expert systems or significant numerical control. On the other hand, however, microchip production requires precise measures, careful regulation of the production environment and checking of the process. As such, it is likely that such production is already under intense mathematical scrutiny, whereas low-tech industries such as textile, infrastructure and construction materials are less so. The company that will be explored as a representative of the low-tech industry, is Company X, a large capacity denim production mill.

2.2.1 Company History

Company X was established as a cotton yarn spinning and weaving firm in 1953. It was restructured in 1986 as a denim production mill and has grown significantly through the 1990s, to reach a production capacity of 45 million meters of denim per year with 1170 employees. Company X is renowned in the world as a leading denim producer, with worldwide customers such as Levis, GAP, Diesel and Replay. Company X has always been an industry leader in utilizing the latest technology, with many machines being developed by companies on site. As such, the machines in the weaving and dyeing range are on the cutting edge of textiles. Yet, even as the machines are advanced and very efficient, mathematical understanding of process behavior - in the dyeing range in particular- is not on par with the technology. Moreover, with the mission of being the preferred denim supplier worldwide, Company X is committed to impeccable quality in its product. As such, human process control measures are in place after the dyeing range.

2.2.2 Denim Production Industry

Company X is on the cutting edge of denim production. While new technologies provide for more efficient and less error prone processes, the basic production process has remained unchanged for decades. Denim goes through six stages of production. The progression through these stages is described in the figure and the stages are as follows:

- Cotton selection and filtering
- Spinning
- Indigo dyeing
- Weaving
- Finishing
- Quality control

2.2.3 Cotton Selection and Filtering

Cotton has many technical properties that affect the end product in various ways. These include stretch, strength, coloring and feel. Various types of cotton are selected and mixed according to their technical qualities, which differ widely according to region. Once mixed, the cotton then goes through a filtering and cleaning process, removing the impurities and weak fibers. What is left at the end of this process is the strong and pure cotton fiber that is used as the core element in denim fabric.

Yarn Spinning

The second stage in the process is the spinning of cotton yarn. The collected fibers are spun with fine tuned machines to produce thick and less dense yarn. This yarn

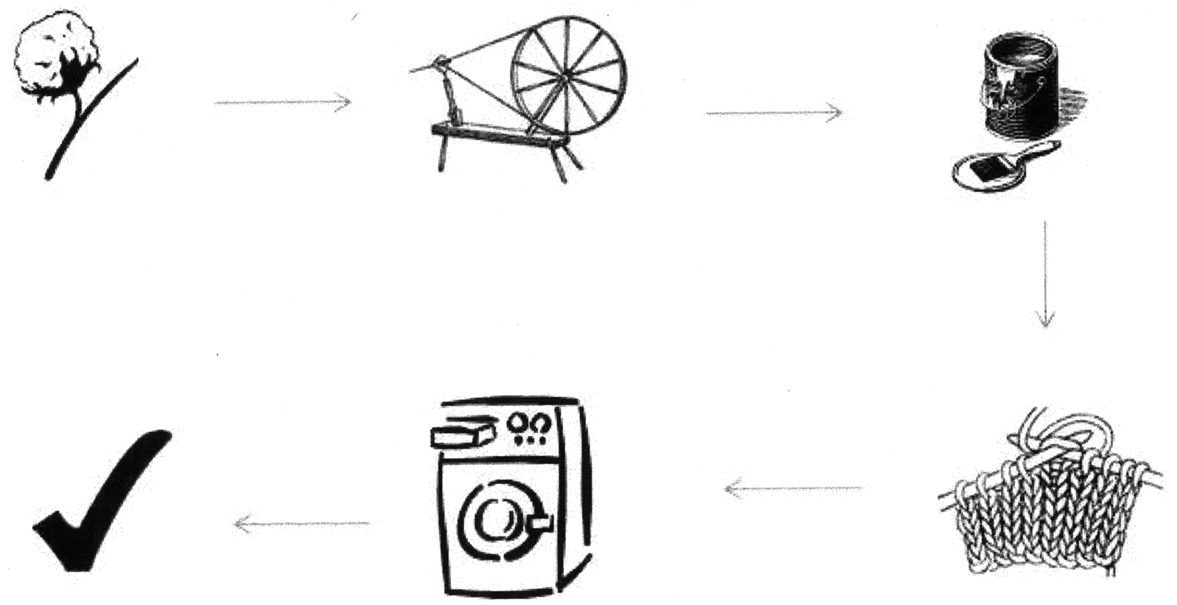


Figure 2-1: The 6 Stages of production in denim manufacturing

is then compressed by a second set of machines to produce strong and dense cotton string.

Indigo Dyeing

The indigo dyeing process is the most crucial, technical and difficult part of the denim production process. Indigo is the signature blue color that we are accustomed to see on blue jeans. This color is given to cotton yarn via the indigo dyeing process. A single long dyeing machine stretches out the cotton yarn and dips it in and out of large tubs of indigo dye, subjecting it to various treatments along the process. This complicated process requires great balance in its parameters (temperature, acidity, indigo flow etc.) and timing. Overall, there are eleven parameters and large tubs that take up to an hour to reach a homogenous state (if, indeed, they ever do). Thus, the ability of a controller to alter the state of the process as well as derive information from it is quite limited. This is one of the primary challenges that was prevalent throughout the research. Further detailed discussion of the process and the complications arising from it will be presented later in this thesis.

Weaving

Weaving is the process of producing denim fabric from cotton yarn. The machines that weave the fabric do so in the age old technique, with the aid of modern technological features such as air-jet technology: Yarn is thrown together via high pressure air jets. While this may seem as the most complicated aspect of denim production, the technique and technology is relatively simple and efficient.

Finishing

In denim, finishing is fashion. This is the stage at which the fabric is washed using different techniques, detergents, chemicals and stones. These products are combined in different ways to give stretch, worn-out and similar effects, anti-shrinkage properties and various color effects. While washing techniques as well as shrinkage protection are utilized throughout the industry, fashion and trends are often set by the different techniques in coating and application of various chemicals for design.

Quality Control

Quality control in Company X is done in various stages. This stage presents an important bottleneck for the mill, as every meter of fabric produced is checked for physical defects by human eyes. There is ongoing research to find an efficient computerized solution to detect defects in the material. A second type of quality control occurs directly after weaving, where the color of the fabric is tested for conformity to specification levels. This quality control is the primary concern of the thesis. Most quality screening and rejection occurs at this stage simply because the indigo dyeing process is volatile and little is understood about the effects of the process on the dye. If we are to limit the rejection of denim fabric based on color quality, we must obtain a statistical and numerical understanding of the process at the multivariate level so as to assure quality in-process, rather than solely relying on a posteriori quality control and wasting material.

2.3 Technical Approach

Our analysis techniques focus on two primary tasks:

- Reducing Dimensionality
- Clustering

Why is it essential that we reduce the dimensionality of the data? There are many techniques that cluster data in higher dimensions. The necessity arises from two factors: the curse of dimensionality and human pattern recognition.

First, higher-dimensional data is usually sparse. Unless there is an extensive collection of data, higher-dimensional space is mostly empty and small features in data points are often missed. Working in lower dimensions eliminates this problem.

Second, humans have the ability to instantly recognize patterns in visual data. To do this however, data must be presented in 2D or 3D. Higher dimensions than this may be represented with various techniques such as color and time (and indeed we will explore such a technique in this thesis) but pattern recognition becomes increasingly difficult. As such, reducing dimensionality- especially to the more meaningful dimensions- provides a great advantage in terms of detecting patterns.

Shaping data so that patterns may easily be recognized by humans is particularly advantageous in our endeavor, as the end-users of the techniques presented here are likely to be process controllers or industry specialists that would prefer ease of use to complicated equations.

Following dimension reduction is the task of clustering. The primary motivation to use discrete clustering, rather than continuous number techniques such as regression is once again related to the end-user. Preemptive quality control defines quality in a binary way: within specifications or not. As such, our interpretation of the data must be along the lines of Go, No-Go decisions. Defining a Go, No-Go decision boundary simplifies the mathematical basis of the task.

Finally, a presentation of findings must also be accompanied by a methodology that process controllers can easily follow without getting bogged down in the cumbersome mathematics. This may be most effectively accomplished by supporting computer software possibly with intuitive visual displays. An important idea towards this end is presented by Chernoff. He claims that humans are most apt at recognizing faces [1]. Moreover, faces have several identifiable features. Therefore, he came up with a method -Chernoff faces- that allows for the mapping and visualization of high-dimensional data via the use of faces: one face for each data point. The location of the point in any one dimension is characterized by the range of a feature. For example, the temperature variable (among the 11 distinct parameters in the indigo dyeing range), may be represented by the slant angle of the eyebrows. The hotter the temperature, the more slanted and angry the face looks.Chernoff faces are a good technique to intuitively recognize patterns and will be explored in this thesis as a simple visual software solution for pattern recognition and process control.

Chapter 3

Methods

This section will include a presentation of the different methods of exploratory data analysis used as well as testing and discussions of performance on simulated data. As the primary task of the thesis, careful thought and testing of these processes will ensure robust or dependable results in the best case. In the worst case, testing will show that the techniques work well, but require a more informative data set.

The four categories of exploratory data analysis techniques along with discussion of visual patterns are included in this section with an exploration of various simulated data distributions. In each case, the techniques we use will attempt to divide the data set into two distinct classes. Half of the data will be used to train - where required- while the other half will be for testing.

3.1 The Common Methods

Every statistical inquiry is likely to include techniques of this group to quickly identify more obvious patterns. Linear regression is among the first methods to be used. It generates a linear formula which predicts the value of a dependant variable, given the values of a set of independent variables. While this technique is not used for binary separation of the data, it is one of the easier and most informative techniques -if successful. As such, it is a good starting point for all exploratory data analysis.

Linear regression attempts to find the minimum least squares function plane through the available data. Once this plane is found, another value that is generated through this method is the R-squared value. This value is a statement in fractions of the amount of explanation in the variation of the dependant variable explained by the independent variables considered. If this number is high, this implies that the variables explain a large amount of the variance. A low number indicated that there are other variables and considerations that effect the dependant variable more than the independent variables considered.

A second form of regression, logistic regression, may be used to represent the binary nature of the problem. Logistic regression enables the use of binary variables, while trying for linear combinations. It determines the probability that a binary event will occur, given the values of the independent variables. This is done by the logistic function:

$$f(z) = \frac{1}{1+e^{-z}}$$

The logistic function output $f(z)$ is a value in the range $[0,1]$. The output $f(z)$ is the probability of an event occuring given the input z. The input to the function is a linear combination of factors:

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_i x_i$$

where $\beta_1 \ldots \beta_i$ are the regression coefficients and β_0 is the intercept

A third method to use in mainstream techniques would include principal component analysis. While PCA may be considered among the more complicated linear techniques, it falls under the usual category as it is used frequently and isn't particularly aligned with our goals. PCA, like regression, generates a linear combination. PCA however attempts to restructure the dimensions of the problem, as orthonormal linear combinations of the original axes. This restructuring is done along the direction of maximal variation [6]. Once this direction is found, the data is projected onto the remaining dimensions normal to this direction and the process is repeated.

Thus the end result of PCA is generating dimensions in a hierarchical order of relevance, in terms of variance. This enables us to throw away the higher order dimensions and analyze only the relevant ones. In that way, unless the projections yield good results, a secondary technique -or visual inspection- would be needed to define the emerging patterns in reduced dimension space.

3.2 Recursive Linear Optimization using Software

While utilizing the computer for extensive blind searches lacks the refinement and certainty of most statistical techniques, it is closest to the human ability to recognize patterns that are intricate enough to be lost in more complicated methods.

As such, a MATLAB software was modified to fit our goals in searching for a pattern. The exploratory data analysis software for MATLAB analyzes data in various forms. One such form is projection pursuit which looks for interesting patters via the projection pursuit technique, with the supplies metric. While most projection pursuit defines interestingness as a departure from normal distribution, and thusly uses the chi-squared metric, our rewrite of the software incorporates the use of clusters: Trying to find the projection with the greatest separation of the two groups, normalized by the standard deviations according to the following.

We have two samples of data, the "go" and the "no-go." We define the "go" sample as

$$\{x_1, \ldots, x_n\}$$

and the "no-go" sample as

$$\{y_1, \ldots, y_m\}$$

where $x_i,\ y_j \in \Re^d$ for all i and j.

Let us define the sample means as:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

and

$$\bar{y} = \frac{1}{m}\sum_{j=1}^{m} y_j$$

Likewise, let us define the sample variances as

$$S_x = \frac{1}{n-d}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^{\mathrm{T}}$$

and

$$S_y = \frac{1}{m-d}\sum_{j=1}^{m}(y_j - \bar{y})(y_j - \bar{y})^{\mathrm{T}}$$

Finally, let us define

$$\mu = \bar{x} - \bar{y}$$

and

$$\mathbf{\Sigma} = \frac{n-d}{n+m-2d}S_x + \frac{m-d}{n+m-2d}S_y$$

Then our best 2-group clustering metric is given by

$$\mu\mathbf{\Sigma}^{-1}\mu^{\mathrm{T}}$$

This metric is useful because it maximizes the separation of the means normalized by standard deviations. As such it ensures groups that are separate and tightly clustered together.

The software picks up a projection into 2D or 3D space at random and refines it to the best value of the metric via rotating through 360 degrees in 10 degree increments. The best value is then recorded and the projection moves to a *neighboring* projection, where neighboring is defined as a tilting of the hyper plane along one dimension. If an improvement is not found in a tilt in either direction, the process moves to another neighboring projection. The method is carried out until all tilts are exhausted with no improvement, or a maximum number of trials are reached.

As many exhaustive techniques of the sort, this method is quick to arrive at local maxima or minima, yet may get stuck locally and is not likely to find the absolute maxima or minima. Yet, for our purposes, while better projections may exist, any projection that does an adequate job at separation is informative enough: It provides us with a basic improvement and better understanding of the model.

3.3 Multivariate Linear Analysis

This method is a stepwise optimized linear projection developed for this research. It attempts to project the optimal separation of two classes of data onto a single dimension. The steps are described below. The advantage of the technique is that it can be applied recursively allowing the projection of the data onto as many dimensions as wanted and, similar to PCA, it finds projections in order of importance and information.

Suppose we wish to reduce the dimension of the measurement space to 1. We can achieve this by using the vector $a \in \Re^d$ and calculate the samples

$$\{a^{\mathrm{T}}x_1, \ldots, a^{\mathrm{T}}x_n\}$$

and

$$\{a^{\mathrm{T}}y_1, \ldots, a^{\mathrm{T}}y_m\}$$

Given the metric as defined in the software technique, our goal is to find the optimal a such that

$$\mu^{\mathrm{T}}a(a^{\mathrm{T}}\Sigma a)^{-1}a^{\mathrm{T}}\mu$$

is maximized. Another way of stating this is that we want to solve the constrained optimization problem

$$\max_{a \in \Re^d} \; (a^{\mathrm{T}}\mu)^2$$

subject to the constraint

$$a^{\mathrm{T}}\Sigma a \;=\; K$$

for some arbitrary but fixed scalar K.

Using a Lagrange multiplier λ, we can rewrite this as

$$\max_{a \in \Re^d} \; J$$

where

$$J = (a^{\mathrm{T}}\mu)^2 - \lambda(a^{\mathrm{T}}\Sigma a - K)$$

Then, we differentiate and set the derivatives equal to zero:

$$\nabla_a J = 2(a^{\mathrm{T}}\mu)\mu - 2\lambda\Sigma a = 0$$

and

$$\frac{\partial}{\partial\lambda} J = -a^{\mathrm{T}}\Sigma a + K = 0$$

Premultiplying the first equation by a^{T}, we get

$$(a^{\mathrm{T}}\mu)^2 - \lambda a^{\mathrm{T}}\Sigma a = 0$$

or

$$(a^{\mathrm{T}}\mu)^2 - \lambda K = 0$$

from the second equation. Thus,

$$\lambda = \frac{(a^{\mathrm{T}}\mu)^2}{K}$$

Substituting into the first equation, we get

$$(a^{\mathrm{T}}\mu)\mu - \frac{(a^{\mathrm{T}}\mu)^2}{K}\Sigma a = 0$$

Then, either $a^{\mathrm{T}}z = 0$ (which can't be optimal since then the objective function, which is nonnegative, would be identically zero), or

$$z - \frac{a^{\mathrm{T}}\mu}{K}\Sigma a = 0$$

so that

$$\frac{a^{\mathrm{T}}\mu}{K}a = \Sigma^{-1}\mu$$

Noting that the magnitude of a is arbitrary, as is the value of K, we choose

$$a = \Sigma^{-1}\mu$$

This is the optimal a if we wish to project d-dimensional measurements onto 1-dimensional space.

Notice that this analytical solution only applies to a 1D projection. Expanding this to 2D requires a recursive application of the same technique and does not provide the ideal 2D projection but rather the best two 1-D projections. As such, the recursive software optimization of the previous section is required to find better 2D and 3D solutions.

The next necessary step is to identify the decision boundary between the two clusters formed by the projection. We will explore two methods to do this: One statistics and one probability approach.

First, if we want to get as much separation between the clusters, the best decision boundary would be equidistant from the two means. Yet, we must also account for the variance in the clusters. Indeed, a higher variance in one cluster should push the decision boundary more towards the tighter cluster. The following method incorporates both the means and variances to create the optimal boundary.

Given two 1-d clusters X and Y, the means of the two clusters are μ_x and μ_y. The variances are σ_x^2 and σ_y^2. We want a line an equal distance of standard deviations away from either mean. This is given by

$$\mu_x + b\sigma_x = \mu_y - b\sigma_y$$

The unknown variable b enables separation with an equal number of standard deviations away from each cluster center. Therefore, we have

$$b = \frac{\mu_y - \mu_x}{\sigma_x + \sigma_y}$$

and the decision boundary is

$$\mu_x + b\sigma_x$$

The second method to generate a decision boundary is to minimize the probability of misclassification. We may do this as follows.

First, let us assume that a decision boundary has been placed.

Let us define N_x as the number of data points in cluster X and N_y as the number of points in Y

Let us further define M_x and M_y as the misclassified data points in X and Y respectively. Thus, M_x is the number of points in X that have been incorrectly classified as points in Y and visa versa.

Now, we must define the probabilities of various events as follows.

The probability of being in X or Y respectively is

$$P_x = \frac{N_x}{N_x + N_y}$$

$$P_y = \frac{N_y}{N_x + N_y}$$

The probability of misclassification of points in X and Y respectively are

$$P_{mx} = \frac{M_x}{N_x}$$

$$P_{my} = \frac{M_y}{N_y}$$

Therefore, the overall probability of misclassification is given as

$$P_M = \text{P(misclassified } X| \text{ in } X)*\text{P(in } X)+\text{P(misclassified } Y| \text{ in } Y)*\text{P(in } Y)$$

$$P_M = P_{mx}P_x + P_{my}P_y$$

Simplifying

$$P_M = (\frac{M_x}{N_x})(\frac{N_x}{N_x + N_y}) + (\frac{M_y}{N_y})(\frac{N_y}{N_x + N_y})$$

$$P_M = \frac{M_x + M_y}{N_x + N_y}$$

The decision boundary, then, is the line that minimizes P_M, the total probability of misclassification. Because the denominator is constant, this is the same as minimizing the absolute number of misclassified points.

3.4 Support Vector Machines

SVM is a classification technique often used in machine learning. The strengths of the technique can be listed as follows:

- Attempts to classify each point into one of two classes
- Provides both linear and nonlinear decision boundaries
- Allows the adjustment of how closely decision boundaries are fit.

A point that we have not stated previously is that in determining a decision boundary for classification one must often decide on the tightness of fit. If we were to mark points individually as members of two separate groups, and drew a line to encompass solely one group, we will have perfectly separated the data. Yet this separation will yield in too specific a fit and is likely to be uninformative. Conversely, too loose a fit is likely to provide information with ambiguous results, and accurate grouping may not be possible.

The SVM algorithm tackles this problem quite elegantly by using margins. While finding a decision boundary on the training data, SVM considers the most closely neighboring data points of the two groups as the margins of the data. These margins -also called the support vectors- determine the decision boundary: The decision boundary is the curve that is equidistant from the support vectors. The algorithm thus tries to find the widest street that separates the two sets of data, with the center of the street being the decision boundary. While ideally we would like perfect separation from SVM, we may also allow for more flexibility and limit SVM in other ways to enable other goals. For example, establishing a minimum width for the margin distances ensures that the fit is not too tight and there is adequate room for closer test data to be classified accurately [5].

The SVM algorithm is as follows: We divide the available data into two groups at random; training and testing data.

We define all training data as

$$\{x_1, \ldots, x_n\}$$

where $x_i \in \Re^d$ for all i

We further define the *class affiliation* (1 for group 1, -1 for group 2) of each data point as

$$\{g_1, \ldots, g_n\}$$

where $g_i \in -1, 1$ for all i

Also note that a hyperplane may be described by

$$\mathbf{w} \cdot \mathbf{x} \; - \; b \; = \; 0$$

where $\mathbf{w}$ is a normal vector to the hyperplane.

In the linear case, we want to choose $\mathbf{w}$ and b so as to maximize the margins. The hyperplanes for this are

$$\mathbf{w} \cdot \mathbf{x} \; - \; b \; = \; 1$$

and

$$\mathbf{w} \cdot \mathbf{x} \; - \; b \; = \; -1$$

The distance between these two hyperplanes is then found as $\frac{2}{\|\mathbf{w}\|}$. Note that the minimum length of $\mathbf{w}$ may be constrained to allow for less tight fits as discussed above.

We must also make sure that

$$\mathbf{w} \cdot \mathbf{x_i} \; - \; b \; \geq \; 1$$

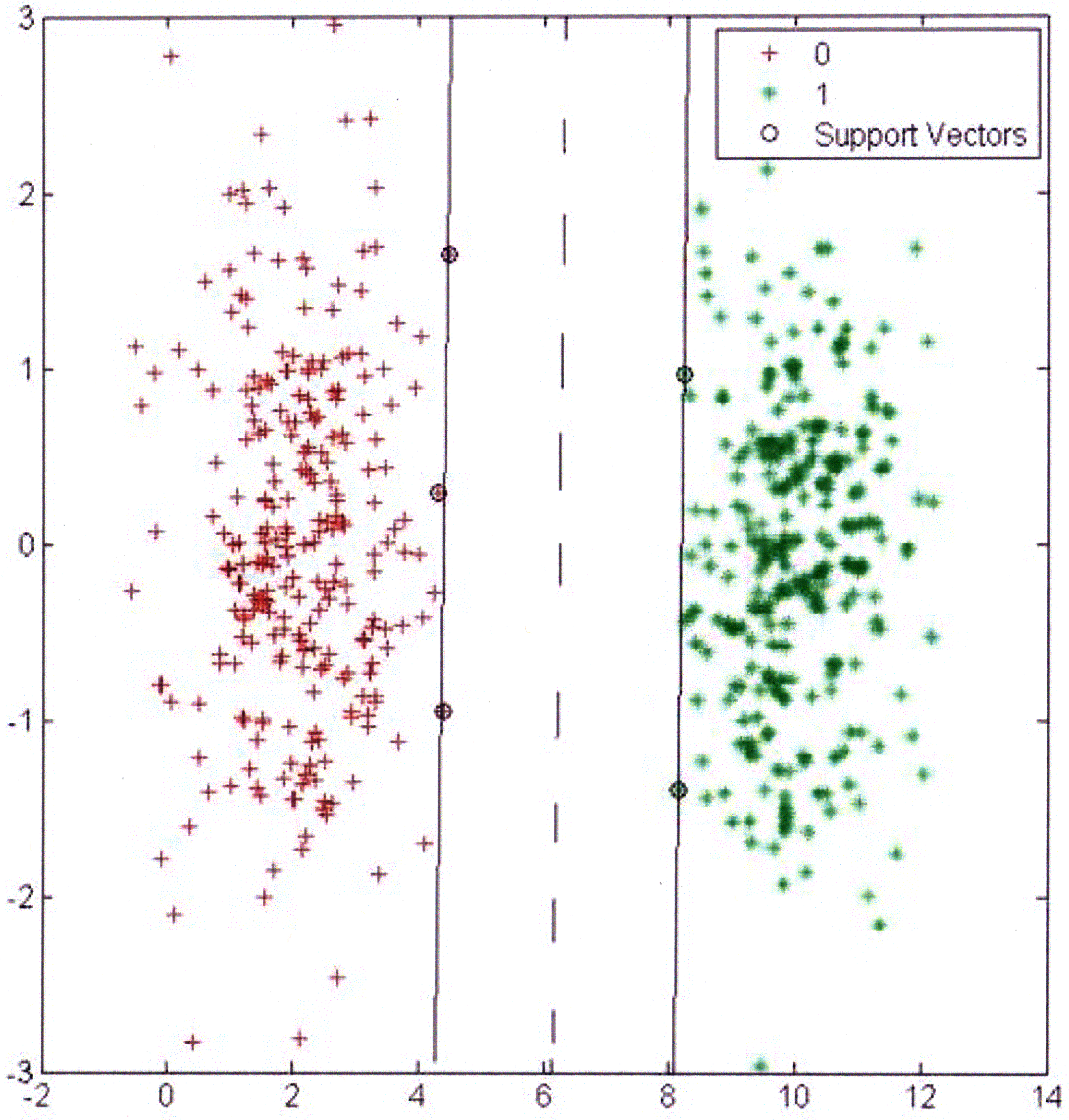


Figure 3-1: Support Vector Machines method: SVM attempts to separate two clusters via a street with maximum margins. The Decision boundary is the dashed line equidistant from the margins

for all x_i with $g_i = 1$ and

$$\mathbf{w} \cdot \mathbf{x_i} - b \leq -1$$

for all x_i with $g_i = -1$

We may write these constraints as

$$g_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1$$

Now, in order to find the maximum width margins, we write the optimization problem

$$\min \|\mathbf{w}\|$$

s.t.

$$g_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1$$

This optimization problem is hard to solve and may be modified in the following manner to use Quadratic Programming techniques

$$\min \frac{1}{2}\|\mathbf{w}\|^2$$

s.t.

$$g_i(\mathbf{w} \cdot \mathbf{x_i} - b) \geq 1$$

Extensions of this linear SVM exist. Most importantly, it may be expanded to polynomial and radial solutions. Also, the optimization constraints may be relaxed for both faster computation time and allowing for near solutions if exact ones are not possible. Furthermore, slack variables may be introduced into the constraints to allow for misclassification of points at a certain cost.

One really powerful property of SVM is its sole reliance on dot products. As the formal presentation outlines, SVM requires only the dot product calculation of classification points with the weight vectors. The power in this -as I will demonstrate- is that it allows mapping to alternative *kernel space* using solely dot products. Explicit mapping of points need not be calculated or even known, but rather may be implicitly obtained through dot products [5]. As such, transforming the points into another kernel only requires that we obtain the same dot product in kernel space, essentially removing the requirement to transform each data point.

Let us define a mapping as

$$\mathbf{x} \rightarrow \Phi(\mathbf{x})$$

Then the quantities required for our optimization are constraints that depend on:

$$\Phi(\mathbf{w}) \cdot \Phi(\mathbf{x})$$

This dot product may be supplied by a function as follows

$$\Phi(\mathbf{w}) \cdot \Phi(\mathbf{x}) = K(\mathbf{w}, \mathbf{x})$$

Therefore, we require only the original vectors **w** and **x** to calculate the dot product in this kernel space.

Possible kernel examples:

Polynomial:

$$K(\mathbf{w}, \mathbf{x}) = (\mathbf{w} \cdot \mathbf{x} + 1)^n$$

Radial:

$$K(\mathbf{w}, \mathbf{x}) = e^{\frac{-\|\mathbf{w}-\mathbf{x}\|^2}{2\sigma^2}}$$

The use of different kernel spaces is a powerful tool. While it is often thought that the curse of dimensionality serves only to diminish information in the data due to sparsity of points in high-dimensional space, the converse may be true. Expanding a data set to higher dimensions or a simpler function space may produce simpler patterns that were previously obscured by complexity. Therefore, we may say that it is not the dimensionality but the complexity that defines the correct approach [5] (projection vs kernel techniques). In our investigation of SVM on simulation data we explore how different kernels change the classification of points.

3.5 Simulations

In this section our methods are tested under various 2-D and 3-D distributions of two classes of data. Each class of data is characterized by two quantities: Mean and covariance matrix. These are used in our random number generator to produce m-dimensional normally distributed data with mean and covariance matrix as specified.

The methods to be tested are logistic regression, multivariate linear method and SVM for 2-D simulations and logistic regression, multivariate linear method, SVM and Exhaustive Computer Projections for 3-D simulations. This is because the projection system used in the exhaustive program projects onto 2-D space. Thus, it does not make sense to find a projection of a 2-D simulation.

We will only be testing these three techniques because the basic techniques such as regression and PCA do not provide decision boundaries either visually or mathematically. They merely serve to capture the directions of maximum variation in univariate data. As such, they may provide valuable information in exploratory data analysis, but not decision boundaries between clusters. Logistic regression is a step above this, in that while it does not present a decision boundary, the probabilistic description may serve as a decision boundary at the equal probability point.

3.5.1 2-D Simulations: The Basic

Non-overlapping samples of normal distributions, separated along axis

This is the most basic simulation of two normal distribution clouds on the plane. The clouds are adjusted so that they can be linearly separated along the x or y axis. Visual inspection shows clearly that this distribution is easily separable linearly. Therefore, we expect that both methods will work well.

The linear method produces a projection onto a single dimension. As such, we must view the points as a histogram of the values they are projected to along a line and the frequency of these incidences. Total separation of the distributions, and the Gaussian shape indicated that both our simulation and method work effectively in separating the two groups.

Table 3.1: Parameters for non-overlapping, vertically separated simulation data

	x_1	y_1	x_2	y_2
μ	1	0	10	0
Σ_x	1	0	1	0
Σ_y	0	1	0	1

Table 3.2: Performance comparison of methods on non-overlapping vertically separated simulation data. Each method does equally well for the given data.

	Log.Reg.	Linear	SVM
Number classified	1000	1000	500
Correctly classified	1000	1000	500
Incorrectly classified	0	0	0
Fraction correct	1	1	1

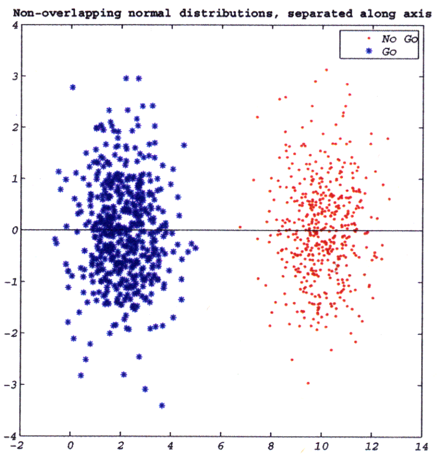
SVM also provides a good solution. While the expectation was that the decision boundary would be a vertical line, the random data provided support vectors that yielded in a slightly slanted separation. However, SVM managed to classify all of the test data accurately after processing the training data. No modification of the kernel or optimization method was necessary for this operation.

Note that while each method accurately classifies 100% of the data, SVM requires half of the data as training points. In this way, the other methods are able to classify more points as they look at the data as a whole rather than separating training and testing data.

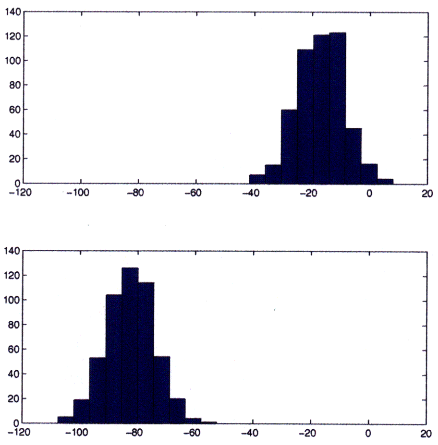
Non-overlapping normal distributions, separated along a diagonal

Visual inspection shows clearly that this distribution is easily separable by a diagonal. Therefore, we expect that all methods will work well.

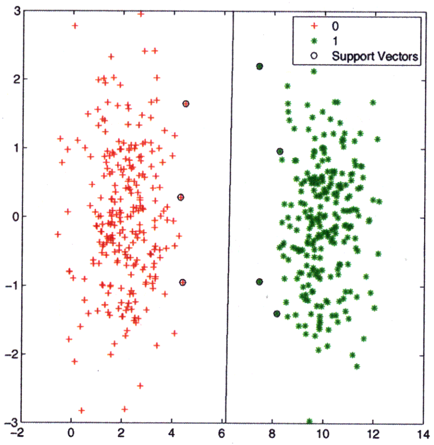
We notice that the methods provide very good but imperfect separation. This, however, is not due to the diagonal but rather because the two clusters have been brought closer together. This was done as a precursor to the next group of simulations which will present overlapping data. From the results obtained, it is quite evident that diagonal vs. vertical separation does not create an obstacle for either method.



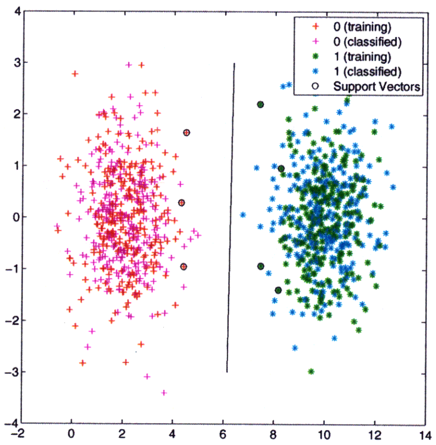
(a) Simulated data non-overlapping, separated vertically



(b) The 1-D linear method projection



(c) SVM separation on training data

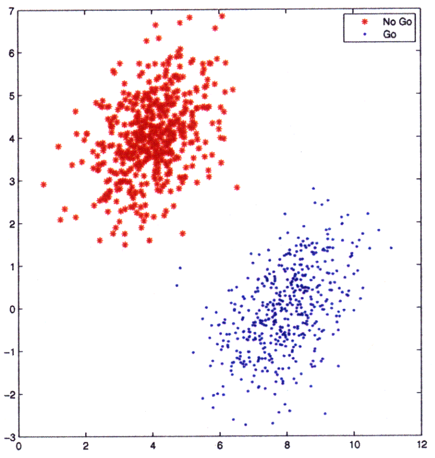


(d) SVM separation with testing data and training data

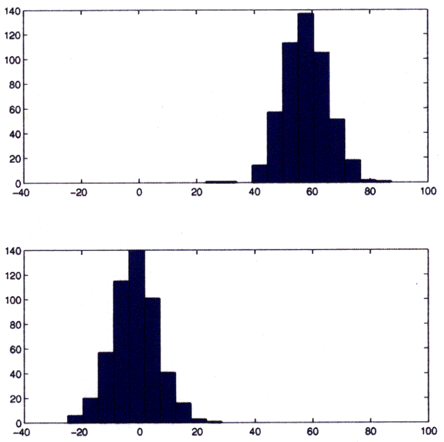
Figure 3-2: Simulated non-overlapping, vertically separated data. Each method does a good job on separating this data.

Table 3.3: Parameters for non-overlapping, diagonally separated simulation data

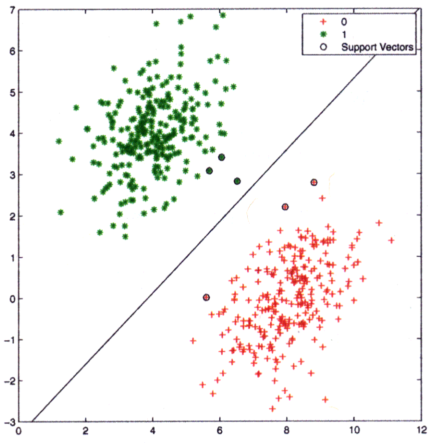
	x_1	y_1	x_2	y_2
μ	4	4	8	0
$\mathbf{\Sigma_x}$	1	0.5	1	0.5
$\mathbf{\Sigma_y}$	0.5	1	0.5	1



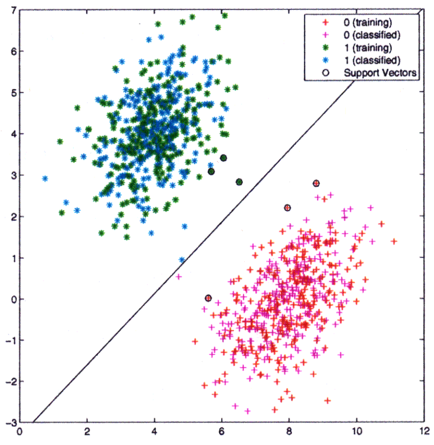
(a) Simulated data separated diagonally.



(b) The 1-D linear method projection



(c) SVM separation on training data



(d) SVM separation with testing data and training data

Figure 3-3: Simulated non-overlapping, diagonally separated data.

Table 3.4: Performance comparison of methods on non-overlapping diagonally separated simulation data.

	Log.Reg.	Linear	SVM
Number classified	1000	1000	500
Correctly classified	1000	1000	499
Incorrectly classified	0	0	1
Fraction correct	1	1	0.998

Table 3.5: Parameters for overlapping simulation data

	x_1	y_1	x_2	y_2
μ	4	4	6	0
$\Sigma_{\mathbf{x}}$	2	0.5	2	0.5
$\Sigma_{\mathbf{y}}$	0.5	2	0.5	2

Table 3.6: Performance comparison of methods for overlapping simulation data.

	Log.Reg.	Linear	SVM	Quadratic SVM
Number classified	1000	1000	500	500
Correctly classified	916	932	460	465
Incorrectly classified	84	68	40	35
Fraction correct	0.916	0.932	0.92	0.93

Overlapping normal distributions

This data involves overlapping points as well as a linear separation of the main clusters. As such, the methods may be expanded to attempt better classifications. While the linear method is fixed as to its adaptability, we may explore various non-linear SVM techniques to capture more than a basic level of linear separation.

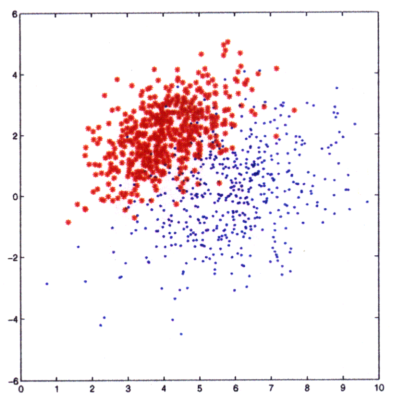
As seen in the figures, a quadratic kernel for the SVM improves classification by 1%. This is not much of an improvement, and in fact given the linear nature of the data may be misleading. Yet, it serves to illustrate the power of SVM Kernel methods as we expand our distributions further.

3.5.2 2-D Simulations: Complicated Distributions

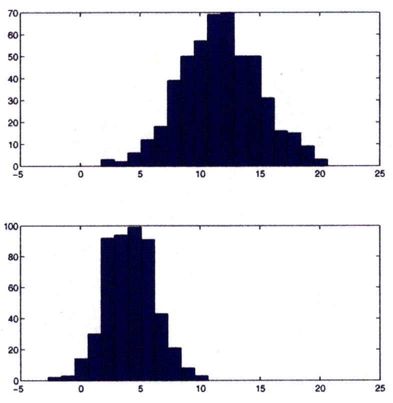
The following cases will have clusters in various forms that are, for the most part, non-linear. Many of these are likely collections from industrial settings.

Circle-in-circle

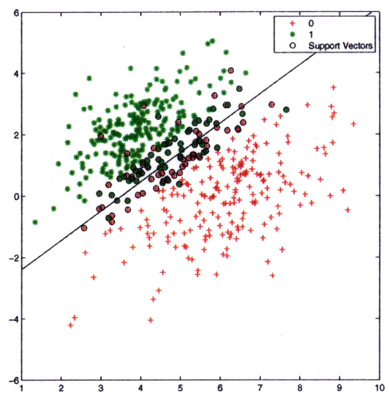
This distribution assumes that one cluster encapsulates the other as a circular crust. This is a likely form for data, in particular if we reject outlier point as No Go. That is, given that Go points are within specifications, all points that lie *outside* these specifications would be No Go points. Thus, the collection of these points -no matter



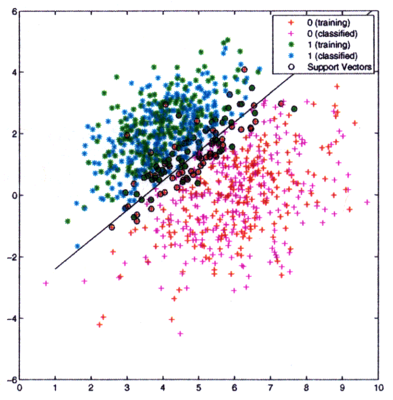
(a) Simulated data, overlapping.



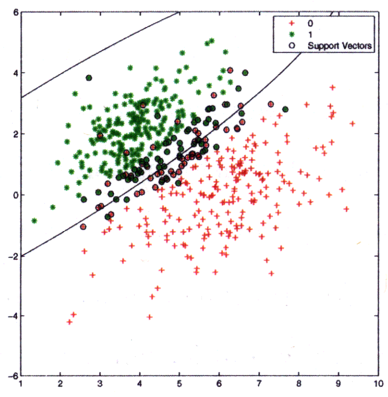
(b) The 1-D linear method projection



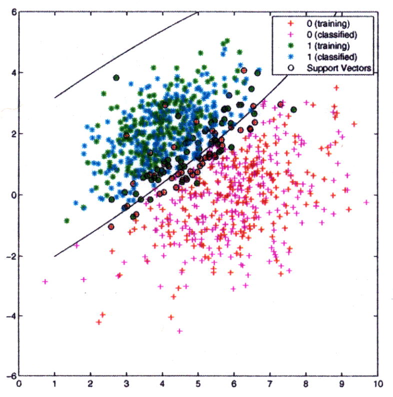
(c) SVM separation on training data



(d) SVM separation with testing data and training data

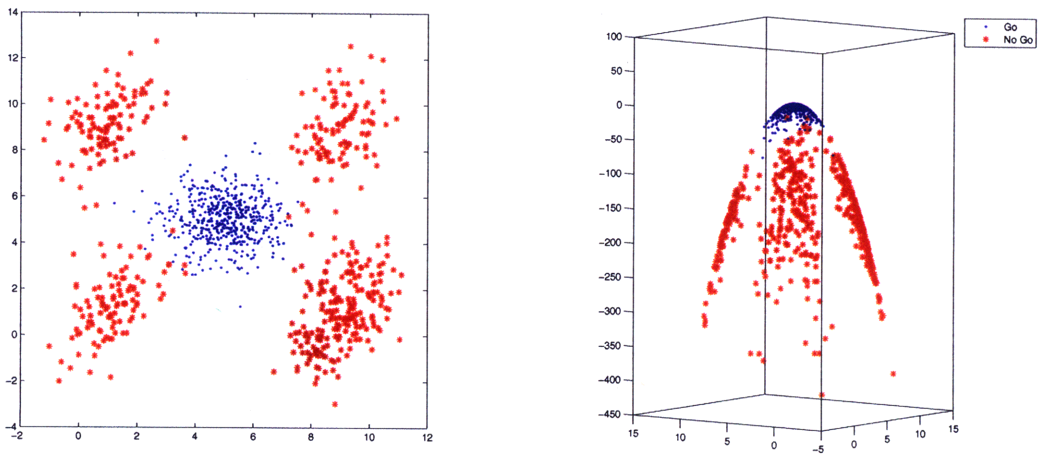


(e) SVM separation with quadratic Kernel on training data



(f) SVM separation with quadratic Kernel on training and testing

Figure 3-4: Simulated overlapping data.



(a) Simulated data, Circle-in-circle.

(b) The Radial Basis Kernel transformation of circle-in-circle data

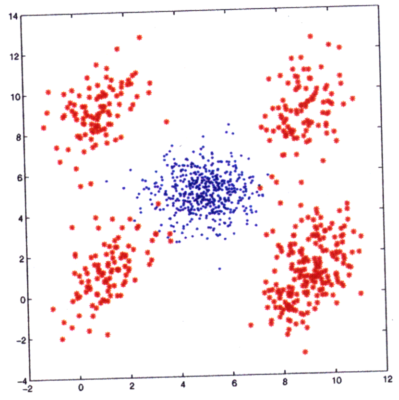
Figure 3-5: Radial basis kernel transformation

Table 3.7: Performance comparison of methods for circle-in-circle data.

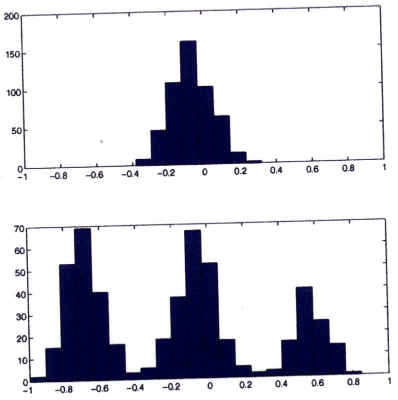
	Linear	Linear SVM	Quadratic SVM	Radial Basis SVM
Number classified	1000	500	500	500
Correctly classified	634	343	494	495
Incorrectly classified	366	157	6	5
Fraction correct	0.634	0.686	0.988	0.99

the direction of their deviation from specifications- will form an outer crust on the inner Go points.

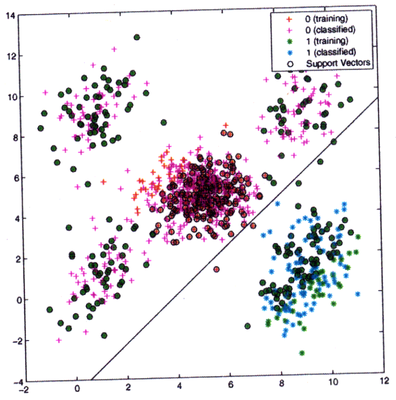
As we may guess, linear methods are unlikely to produce reliable decision boundaries as the boundary we wish to impose is likely to be circular. The power of SVM Kernel methods becomes apparent in this type of distribution. Namely, a radial basis kernel or a polynomial kernel is much more likely to wrap around the inner circle as a decision boundary. The visualization of the kernel transformation is shown below. The linear separator for the radial kernel space becomes a radial separator in the original space.



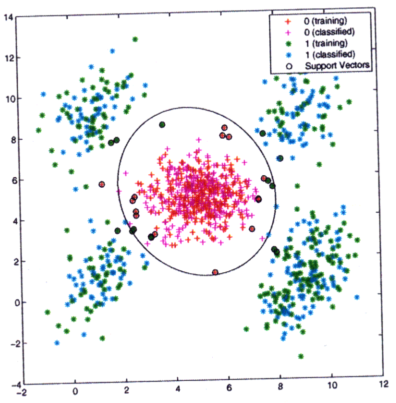
(a) Simulated data, Circle-in-circle.



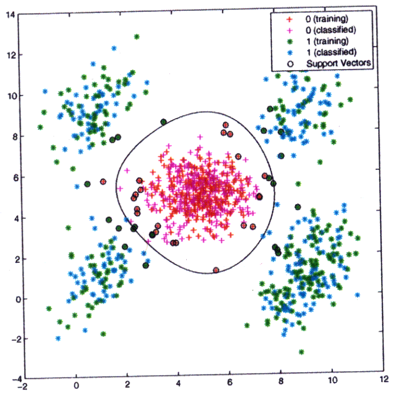
(b) The 1-D linear method projection



(c) SVM with linear Kernel



(d) SVM with quadratic Kernel



(e) SVM with radial basis Kernel

Figure 3-6: Simulated Circle-in-circle data.

Chapter 4

Case Study: Company X

So far we have developed the technical approach and methods that are to be applied in industrial cases. As part of this thesis and research project, we endeavor to achieve our objective: To preemptively identify faulty end product via in-depth analysis of process data. As discussed previously this will be done in three steps:

- Obtaining and processing data
- Performing Exploratory Data Analysis
- Developing decision strategies

We systematically applied the three stages necessary for our objective, to Company X. The following chapters will present our approach, results and discussion.

4.1 The Indigo Dyeing Process

The process for which we are applying exploratory data analysis may be collectively described as the coloring of denim fabric. Many different colors may be used in coloring. In order to focus the study, analysis will be based on the pure indigo coloring process.

The collective coloring process consists of two main parts; dyeing and washing. Both of these parts play a role at creating the final outcome color. Therefore, it is evident that the study must deal with different sets of parameters in both parts in relation to the collective outcome parameters.

If we are to produce meaningful correlations between input parameters and outcomes, we must gain an understanding of the process and how it relates to data obtained from the plant. The following sections will describe the characteristics of indigo dye, the process and the data derived from the process. I will then conclude with the difficulties, approach and goal of the study in relation to the process and data.

4.1.1 Indigo Dye

Indigo is a unique dye used in the coloring of denim. Unlike many different types of dye, indigo is known to be flexible in color and often unpredictable. These same qualities that make it flexible also make it difficult to achieve consistency in coloring. For these reasons there exists an entire field of indigo engineering, dedicated to the methodology and production of various colors and characteristics of this versatile dye.

Several tools are available for the indigo engineer to manipulate the color characteristics of indigo. First, and most relevant to our study, is the dyeing process in which many parameters contribute to the outcome color. Second, the post-dye washing of denim fabric allows for different characteristics to be imbued into the dye giving it different appearances. In general, we may say that the dyeing process provides the first order color properties (blue, blue-green black etc.) and the washing process provides second order properties (brightness, shine, ageing and wearing effects).

While indigo engineers are able to achieve the colors and effects they desire through previous knowledge, experience and trial and error, there is great variance in the end results often falling out of standard. The aim of this case study, then, is to facilitate the production of consistent coloring and develop a higher level of understanding between the correlation of parameters and outcomes. In this way, the thesis provides a mathematical approach to solve a low-tech industrial problem. As such, the indigo dyeing process at Company X provides an excellent test.

4.1.2 The Coloring Process

Below we outline the general indigo dyeing process. The process is complicated in many ways, primarily in the interchanging units of production and the bundling of product. Below are a few of the terms and their size:

Work order The biggest unit containing a set of coloring characteristics and process parameters - Over 10000 meters of yarn

Line: A line goes through a single dyeing machine in one go (although consecutive lines often follow directly as well) - 1000 to 2000 meters of yarn

Ball: Dyed yarn is spun onto balls for transport - 100 to 200 meters of yarn

As stated there are two components to the coloring process; dyeing and washing. Each coloring process starts with a work order that specifies a length in tens of thousands of meters of yarn and a color for the yarn to be dyed. Also specified in the work order are the variance standard and rejection criteria for the coloring. The work order is the largest unit of production for the dyeing process. It contains a single set of target color values and a prescription of process parameter values to reach this color.

The work order is then split into lines for coloring, where each line is approximately 1000 to 2000 meters of yarn that goes through the dyeing machine in one session. After a line of yarn is dyed, it is spun into balls of yarn which are actually cylindrical containers each containing around 100 to 200 meters of dyed yarn.

Dyeing occurs as yarn is continuously dipped into and out of very large tubs of indigo dye. It is further treated by chemicals and steam along the way. The process is controlled by the specified prescription in the work order. Control is exerted over 11 parameters via a computerized system. It is essential to note that the procedure for altering the parameters is very slow. The dyeing range consists of very large tubs of dye that take up to an hour to homogenize. As such, the effect of any controlling action takes up to an hour to be homogenously distributed throughout the tubs. Therefore, yarn dipped at a certain point is not subject to the same conditions as yarn further down the line. Moreover, the point of measurement taken for the controlling system is placed arbitrarily and does not necessarily reflect stable and homogenous conditions in the tub.

After dyeing, the balls of yarn are then washed according to washing prescriptions. The washing process parameters are very approximate and are often changed manually via controllers without accompanying records. After this washing, the yarn is assumed to have the appropriate color characteristics and is sent to weaving. Different washing prescriptions are often used within the same work order as the quality of the color and its conformity to standards is only checked after the end product denim is produced.

Quality control is achieved as follows. Every work order for dyeing is accompanied by a master sample. This master was previously developed under the same factory dyeing and washing conditions and selected by the customer as the target color for the denim. This master sample specifies for the dye and washing range both the target color values and the necessary process parameters and washing prescription to get to this target.

After the end product is produced, the balls of yarn are now balls of denim. A sample patch of denim from each ball is taken and affixed to the master sample to form a blanket. This blanket thus contains samples from various points in time in the process. Each sample is then compared with the master according to the output parameters outlined below. If deviations in parameters are acceptable the balls are shipped to the customer, if not they are recycled into the washing stage until they become acceptable for shipping to a customer in Company X's portfolio or sold on the spot market.

4.1.3 Process Parameters

OUTCOME PARAMETERS:

The outcome parameters of the coloring process consists of three variables collectively defining the quality of color in the fabric; L,a,b.

L: Dark-Light fabric and varies from 0 (darkest) to 100 (lightest)

a: Red-Green fabric that varies from 0 (green) to 100 (red)

b: Blue-yellow fabric varying from 0 (yellow) to 100 (blue)

The three variables constitute the standards in judging the coloring quality. The master sample sets specific values for L,a,b and the other samples in the blanket are measured for deviation (Δ) from these set values.

INPUT PARAMETERS - DYEING:

The dyeing range has many input parameters, each of which greatly effect color. The parameters considered are:

Indigo Concentration: The amount of indigo in the dye baths in grams per liter

Hydro Concentration: Amount of water in the dye baths in grams per liter

Caustic Concentration: The caustic concentration in the baths in grams per liter

pH: The pH value of the dye mixture in the baths

Hydro Flow: The flow of water into and around the dyeing range in liters per minute

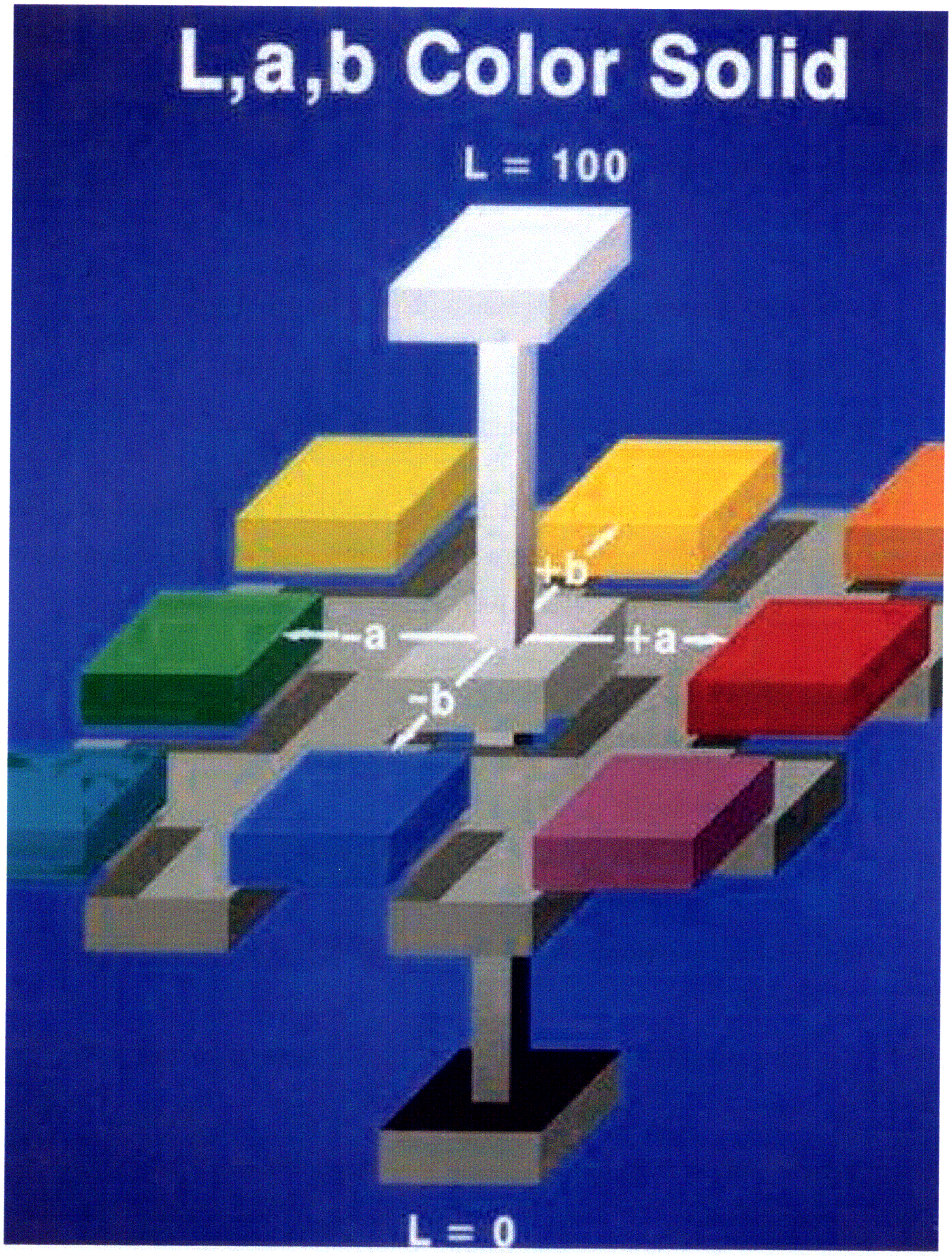


Figure 4-1: Color Quality Chart: L,a,b are the three variables which quantify and characterize indigo color quality

Indigo Flow: The flow of indigo into and around the dyeing range in liters per minute

Pre-processing Temperature: The temperature of washing and processing prior to coloring in centigrade

Indigo dye temperature: Temperature of dye mixture in baths in centigrade

Speed: The speed at which the yarn travels through the system in meters per minute

Humidity: The amount of humidity in and around the dyeing range

Conductivity: The conductivity in the baths due to dissolved inorganic salts

INPUT PARAMETERS - WASHING:

The washing process is carried out according to a fixed prescription. While the parameters in the process are likely to vary, data on this variance is not available. This is mostly due to the nature of the data. For example, stones are often used as part of the washing process but the exact amount, size and distributions of stones in the washing machine is subject to great and unmeasured variability.

Prescription Code: The prescription code is the only recorded parameter in the washing process.

4.1.4 Target Analysis and Issues

According to the indigo engineers at Company X, there are three main forms of unexplained variance of color outcome. Each of these are possible targets for exploratory data analysis and the reduction of variability.

1) **Same Prescription, same blanket:** The samples on the blanket have a certain Δ from the master sample despite having been through the same dyeing and washing procedure. This variance may be analyzed and explained by variance of dyeing process measurements.

2) **Same Prescription, different blanket:** It can be seen that different blankets of the same prescription have different master sample values after washing as well as different delta patterns. This implies that although the same procedures are carried out for two blankets formed of the same balls of denim, an additional variance is introduced in washing.

3) **Different Prescription:** When comparing two different blankets from two different prescriptions, it can be seen that the Δ from the master sample of the other samples follow a certain pattern in one blanket and a completely different pattern in the other. This difference may be explained through dyeing parameters as well as the unique interaction of these parameters under different prescriptions. Identifying such interactions may allow for the reduction of variance. An example of this variance may be that the trend of samples may be an increasing $\Delta(L)$ for prescription 1 and a decreasing $\Delta(L)$ for prescription 2. If prescriptions did not uniquely interact with the dyeing process then the $\Delta(L)$ trends would have been in the same direction. However, controllers have informed us that this is often not the case. Therefore, it is important to understand how the washing prescription can change the direction of deviation in a given L,a,b value.

Given the availability of data, the case study was aimed towards the analysis of 1) and 3). The primary analysis holds prescriptions and washing constant while performing analysis on the dyeing process parameters as they apply to Δ variances. The secondary analysis was meant to identify relations of specific dyeing process conditions as they apply uniquely and differently to different prescriptions. However, as we will discuss in the following sections, the available data prevented most objective analyses.

4.2 Data

The first step in our three step objective was to obtain and process data into a form in which we may perform our exploratory data analysis. Below is a description on how the data was obtained and modified to achieve this as well as a discussion of shortcomings.

4.2.1 Sources

The data received from Company X is contained in three main files as follows. Sample sections of each may be found in the appendix:

Process Measurements: This is the data that contains the parameters during indigo dyeing

Output Results: This is the color result data after dyeing and washing, at a point which the fabric is ready to be shipped

Line Data: This is additional data used to linked the process parameters with the outputs. It contains lengths of all lines of all work orders

I shall refer to process parameters as *input*, output results as *output* and line data as *linking data.*

4.2.2 Approach

The data as given must be manipulated in order to be ready for exploratory data analysis. The task is to accurately match the process parameters with the results they produce (the input with the output). Moreover, the goal is to derive meaning from the input parameters that will identify parameters that yield acceptable outputs and unacceptable outputs.

There are several challenges that relate both to the physics of the system and to the presentation of available data.

Matching input and output:

The input and output data do not correspond directly. That is, input measurements are taken every hour and recorded with a time value, while output is analyzed in blankets spaced evenly among different balls of fabric and recorded with a length value, with no particular time spacing.

The essential variable used to tackle this problem is the meters/minute recording of the speed at which a line of fabric goes through the process. This, multiplied by the amount of minutes, accurately calculates the length for which a particular process parameter recording is valid for. From this point, the method has two versions that match this length value with the length value in the output data.

First, the output data contains the length for each roll of fabric. Moreover, only one output measurement is taken from a single roll of fabric. Thus, we may associate each sample with a certain length in the process assuming that the balls of fabric are continuous products coming from each line. While this assumption seems valid, it is also possible that there are missing balls of fabric that would offset the matching of lengths in the production line with the lengths of end product.

Second, and more effectively, we may use the linkage data that presents the length of each line in production. From this information we may extrapolate which line contains what length range. This information links the line numbers to the input data via the meter/minute measurement. Moreover, output results contain the line number that each result is a part of. Thus through the line data as the linkage, an association between input and output is achieved.

Input data can be accurately estimated through meters/minute data as to which measurement corresponds to which line. As each measurement corresponds to roughly 1 line, we can match each record to one or two lines.

Granularity:

Granularity is a significant issue in the data. Input parameters are measured once every hour, and therefore correspond to more than one of the output records. This is not due to insufficient bookkeeping but is an inherent problem in the process. Because each dye range contains large containers of dye, the process parameters homogenize at a very slow pace (a modification to the input parameters takes approximately an hour to be assumed to be homogenously distributed).

Because each input corresponds to several outputs, it is often the case that one set of input parameters have outputs of greatly varying measures. This results in many points that are both good and bad data points. As such, grouping and clustering analysis to separate good results from the bad becomes challenging.

There are a few ways to tackle this problem. First, rather than using a binary measure of end result as good or bad, we derive a continuous variable representing the goodness of output. Second, for a given set of results representing the same input parameters, we may take the majority result as the overall result for the parameters. Indeed, bad results are much rarer than good results. As such, if a given set of inputs produce even one bad output, they may be classified as bad inputs.

Note that while granularity mismatch is a concern for creating unique data points, it is an advantage in matching inputs and outputs. This is because there is a large margin for error as the input measurement is valid for a large range of lengths due to the difference in granularity.

Cluster Classification:

Another task required to prepare the data for analysis is classification. As previously motivated, using clustering techniques on classified data is the best approach to understanding the data. This is because, rather than trying to establish direct relationships between inputs and outputs, our target industries are primarily concerned with products being within specifications or not. As such, the quality control decision is often binary with a clear decision boundary.

We attempt to establish this boundary, which is normally a function of the outputs, as a function of the inputs. The linear and SVM clustering techniques require that we divide the data into two clusters, as Go and No-Go.

There are several approaches towards this end. First, the obvious method would be to use master sample specifications as a cutoff point for go no-go values. The problem with this is that the specifications in the sample are flexible. That is, while prescriptions and ideal color values may be the same under two work orders, each order might have a different tolerance for error entirely due to the customer. These arbitrary differences will create inconsistent clustering throughout our data.

The second approach to take is similar to standard quality control techniques. 3σ quality control assumes that if products are three standard deviations or more away from the mean, the process is faulty and must be checked for systematic errors. If we assume the same as being true in indigo dyeing, our classification would be to mark all inputs that correspond to greater than three standard deviations of variance as No-Go. This deviation is measured as the difference from master sample values for each work order. Thus, we may say:

For a given work order with N data points, let $\mathbf{x_i}$ be the vector of input parameters associated with $\mathbf{y_i}$ the vector of output parameters such that the inputs are

$$\{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$$

and the output vectors as Δ's from the master sample are

$$\{\mathbf{y_1}, \ldots, \mathbf{y_N}\}$$

Our matching function $\chi(\mathbf{x})$ maps the inputs to the outputs as follows.

$$\chi(\mathbf{x_i}) = \mathbf{y_i}$$

The mean and standard deviation of the output L,a,b Δ's are given by

$$\bar{\mathbf{y}} = \sum_{i=1}^{N} \frac{\mathbf{y_i}}{N}$$

$$\Sigma_{\mathbf{y}} = \sqrt{\frac{1}{1-N} \sum_{i=1}^{N} (\mathbf{y_i} - \bar{\mathbf{y}})^2}$$

At this point the standard deviation classification may vary according to a few considerations. First, the number of standard deviations may be varied. As 3σ is the most common discriminator between acceptable and unacceptable results, we will use this in our research. Second, the classification may be made for one dimension of the vector $\mathbf{y_i}$ such that if one dimension is more than 3σ away, the vector $\mathbf{y_i}$ may be considered No-Go, or we may require all dimensions to deviate such that all of $\Sigma_{\mathbf{y}}$ is more than 3σ away. We may define the distinction as:

Let α be Go points and β be No-Go points. Then α and β are given by:

$$\forall i, j \ \frac{\mathbf{y_i}_j - \bar{\mathbf{y}}_j}{\Sigma_{\mathbf{y}_j}} > 3 \longrightarrow (\mathbf{x_i} \in \beta)$$

$$\forall i, j \ \frac{\mathbf{y_i}_j - \bar{\mathbf{y}}_j}{\Sigma_{\mathbf{y}_j}} \leq 3 \longrightarrow (\mathbf{x_i} \in \alpha)$$

for classification through only single dimensions, and:

$$\forall i, j \ \frac{\mathbf{y_i} - \bar{\mathbf{y}}}{\Sigma_{\mathbf{y}}} >_e 3 \longrightarrow (\mathbf{x_i} \in \beta)$$

$$\forall i, j \ \frac{\mathbf{y_i} - \bar{\mathbf{y}}}{\Sigma_{\mathbf{y}}} \leq_e 3 \longrightarrow (\mathbf{x_i} \in \alpha)$$

for classification through entire vector. Note that the division and inequality above is elementwise.

Our approach will be primarily a 3σ deviation and single dimension classification.

4.2.3 Methods

Assembling Unified Data

Microsoft Excel and Visual Basic were used to match the input and output data according to the two techniques described (using ball length data and line length data)

The code for the roll of fabric lengths follows the following algorithm:

```
(1) For each row of output record,
        For each work order,
(2)         If current input meter count > current total ball meter count
              Copy current input parameter row to current output row
              Advance to next output row and loop to (1)
            Else
              Advance to next input row
              increment current input meter count and loop to (2)
```

The Line of Fabric lengths matching method requires two separate pieces of visual basic code. First, we must identify the starting and ending line for each input parameter record. Also, we must calculate the fraction of each line that a given parameter is part of. Second, given the start, end and fraction data, we must match the output records to each of the input records. The output records are assumed to be evenly distributed along each line.

The first task is accomplished in Visual Basic as follows:

```
For each row of input parameters,
    For Each work order,
        Reset current line number and meters-to-process (mtp) variables
        Get the meters per line (mpl) and meters of production (mop) of given row
        Record start line number
        If $mtp - mop < 0$
```

```
            Increment current line number, record end line number
            Fraction of end line produced = (Mtp-mop) / mpl
        Else mtp = mtp - mop
        Increment current line number
    Loop to start
```

The second task involves different cases of where the input parameters lie in relation to lines. A single record might encompass an entire line, start a line but not finish it, or finish a line and start the next. As such the code ensures that each case is covered while assigning the input to the output.

Classification

Classification of data points to prepare for clustering takes into account the techniques described in the approach as well as the shortcomings in the data: granularity and redundancy. First, as discussed, the input and output granularity don't match. One input corresponds to many output values and as such the same inputs may be classified as both Go and No-Go due to differing output values. Second, the data is highly redundant. In fact, while an initial sample of the data contained over 3000 input data points, only 550 of these points were unique. This also may cause the classification of same input values to different outputs and classifications.

The methods used to tackle these issues in conjunction with the classification technique are twofold: Homogenize or cascade. Homogenizing the data involves making one decision -Go or No-Go - for each unique input data. This ensures that we will not have ambiguous points to cluster. In terms of the decision, given our 3 sigma classification boundary, No-Go points are considerably rarer than Go points. In fact, if any set of inputs yield in a No-Go point; we assume that this set is a bad set because it is a rare case that produces such outlier outputs.

The cascade technique attempts to separate the data further by generating several classes rather than a binary Go, No-Go. Data points are classified according to the number of standard deviations they are away from the mean. After cascading the points as such, we take the highest standard deviation output result and generalize it to the entire set of identical inputs. In a way, the cascade technique may be thought of as an expansion of the homogenizing method but is middle grounds between individual classification of points and completely reclassifying a great number of points.

The algorithm for both techniques involves going through each output produced by the same input and associates the entire group to one class. This algorithm eliminates ambiguity due to granularity but does not eliminate redundancy. Redundancy is a difficult issue to tackle as each data point needs to be compared to the entire set of data points, creating an algorithm that grows exponentially. A basic technique would generalize to one class -Go by default and No-Go if it is ever found- by looking through the data and eliminating all points of the other class.

4.3 Exploratory Data Analysis

The second, and most important, step in our three step objective is to perform exploratory data analysis on the data using the various methods explored in the previous chapter. We present this using the same categories discussed in the methods.

- The usual
- Exhaustive software
- Multivariate linear analysis
- Support Vector Machines

Essentially, this approach ensures a look at the data in increasing detail. First, we provide a general look using common techniques such as regression. Second, we attempt non-specific and non-precision computational techniques. Following this, we employ mathematical techniques of increasing complexity and pattern recognition ability.

Table 4.1: Mean and Variences of input parameters

	Ind. Conc.	H_2O Conc.	Caustic	pH	H_2O flow	Ind. Flow	Temp
Mean	1.3228	0.3929	0.8375	12.5183	1.0181	1.4404	39.5226
Varience	0.0093	0.0057	0.0252	0.0027	0.0723	0.881	206.85

	Ind. Temp	Speed	Humidity	Conductivity
Mean	26.8985	20.0987	6.3488	54.0343
Varience	0.3045	0.1891	3.3548	65.6837

An essential part of this process is recognizing that humans are the most efficient pattern recognizers. Therefore, our efforts will be primarily based on reducing dimensionality in a way that preserves the most amount of information we want, in order that we may visually recognize patterns. Thus, the task is to rely on statistical techniques that facilitate easy visual pattern recognition. Much of the analysis will, therefore, be presented visually and comments will be based primarily on the visual appearance of graphs.

4.3.1 The Common Methods

In order to understand the data more thoroughly we must first explore individual parameters and their relationships to other parameters. The aim of this is to identify particular elements of variance as well as any patterns that are prevalent or obvious.

A Look at Means and Variances

As seen in the table, the parameters with the most amount of variance are Indigo Flow, Temperature, Humidity and Conductivity. In fact, the other parameters have negligibly small variance from their mean values. As such, these four parameters may be thought to be primarily responsible for variations in outputs.

2-Dimensional relationships of parameters

Looking at patterns also involve the 1-to-1 correlations of the parameters. The following figure is a comparison of the parameters in this way.

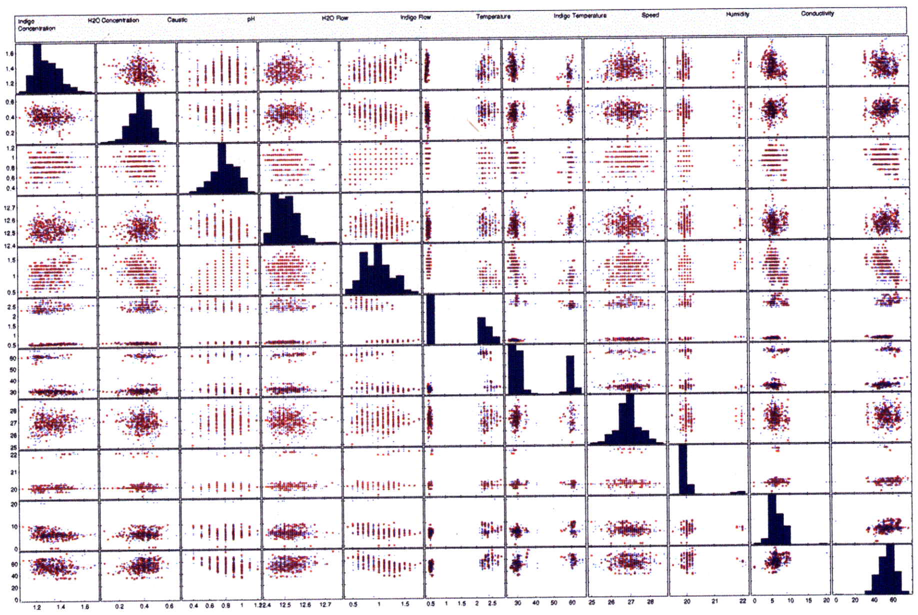


Figure 4-2: 1-to-1 Comparison Graphs. Individual histograms shown along diagonal.

Table 4.2: R^2 values of linear regressions

	L	a	b
Deltas	0.1828	0.1602	0.0565
Absolute Values	0.0541	0.0534	0.0405

Linear Regression

In order to determine the possibility of a linear relationship between the inputs and outputs, we performed linear regression on the inputs for each output L,a,b. As expected, the outcome proved that a linear relationship did not exist, given the low R^2 values.

We consider both the deltas and the absolute L,a,b values in the regression, neither of which provides high R^2. This result also discourages linear projection methods to obtain results. It is likely that non-linear patterns - if any - would emerge from the detailed analysis.

Principal Component Analysis

Another method often used in multivariate analysis is Principal Component Analysis. This technique will provide us the most basic method of projecting onto lower dimensions and visualizing data. PCA enables us to select the axes with the most amount of information in terms of variance. It transforms the data onto the same number of dimensions as the original, but with projection axes that contain the most amount of variance to the least. Thus, we may throw away the lower axes and still preserve the most informative axes.

As seen in the figure, this yields two clusters, however given our classification of the data No-Go and Go points are very evenly divided among the two clusters. As such, PCA does not yield in an informative result for our purposes.

4.3.2 Projection Pursuit via Recursive Software Optimization

The second stage of analyzing the data involves non-specific software analysis. As outlined in the methods, software analysis involves an iterative procedure in which the software attempts to find the projection with the most optimal value of a specified metric. We had developed the metric as follows:

$$\mu\Sigma^{-1}\mu^{\mathrm{T}}$$

Running the software over 30 trials on our cluster separation metric, the figure below shows the optimal 2-D projection found. As can be seen, the projections yield no separation of the Go, No-Go points.

4.3.3 Multivariate Linear Analysis

Multivariate linear analysis attempts to find the best 1-d projection that separates the two clusters. The technique may be used to find n-d projections via recursively applying the same method to the subspace that is orthogonal to the 1-d projections.

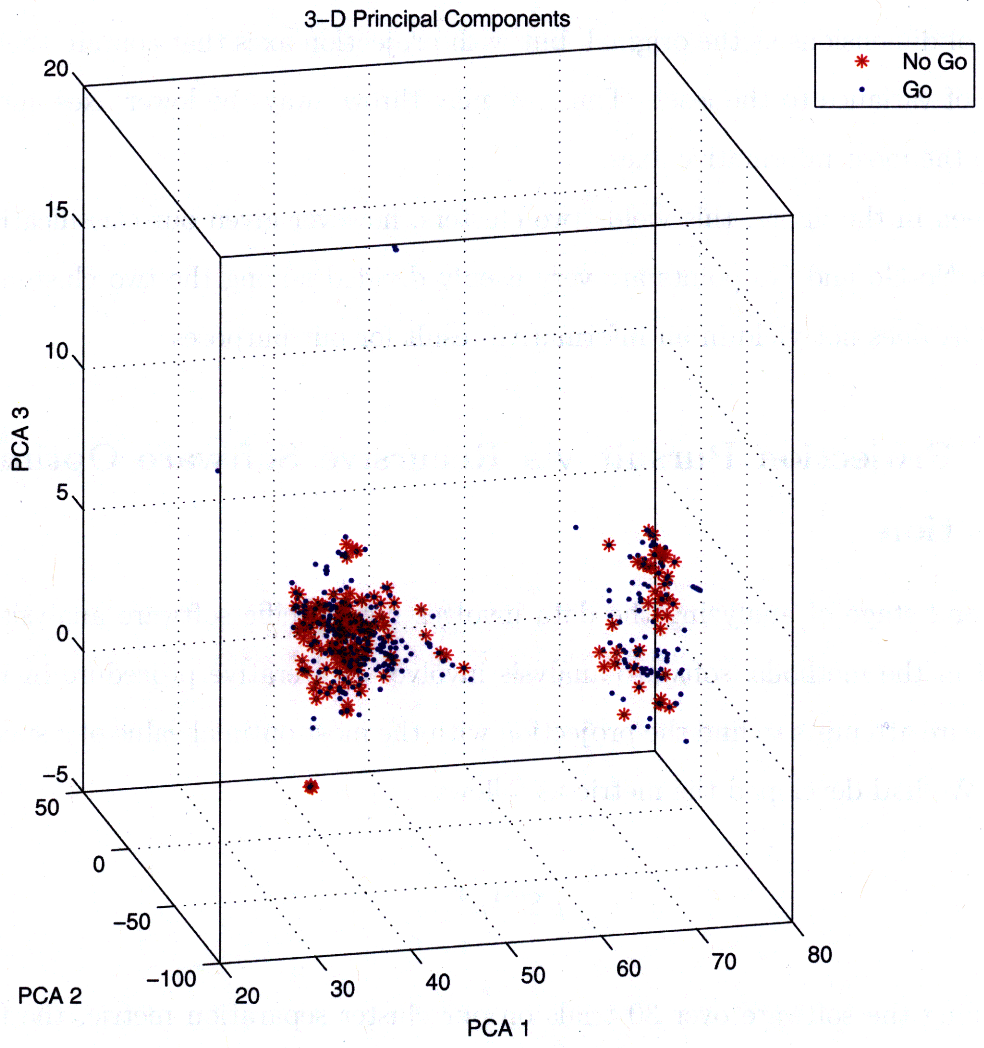


Figure 4-3: Principal Component Analysis projecting the data onto 3-D space. The clusters have similar amounts of Go and No-Go points

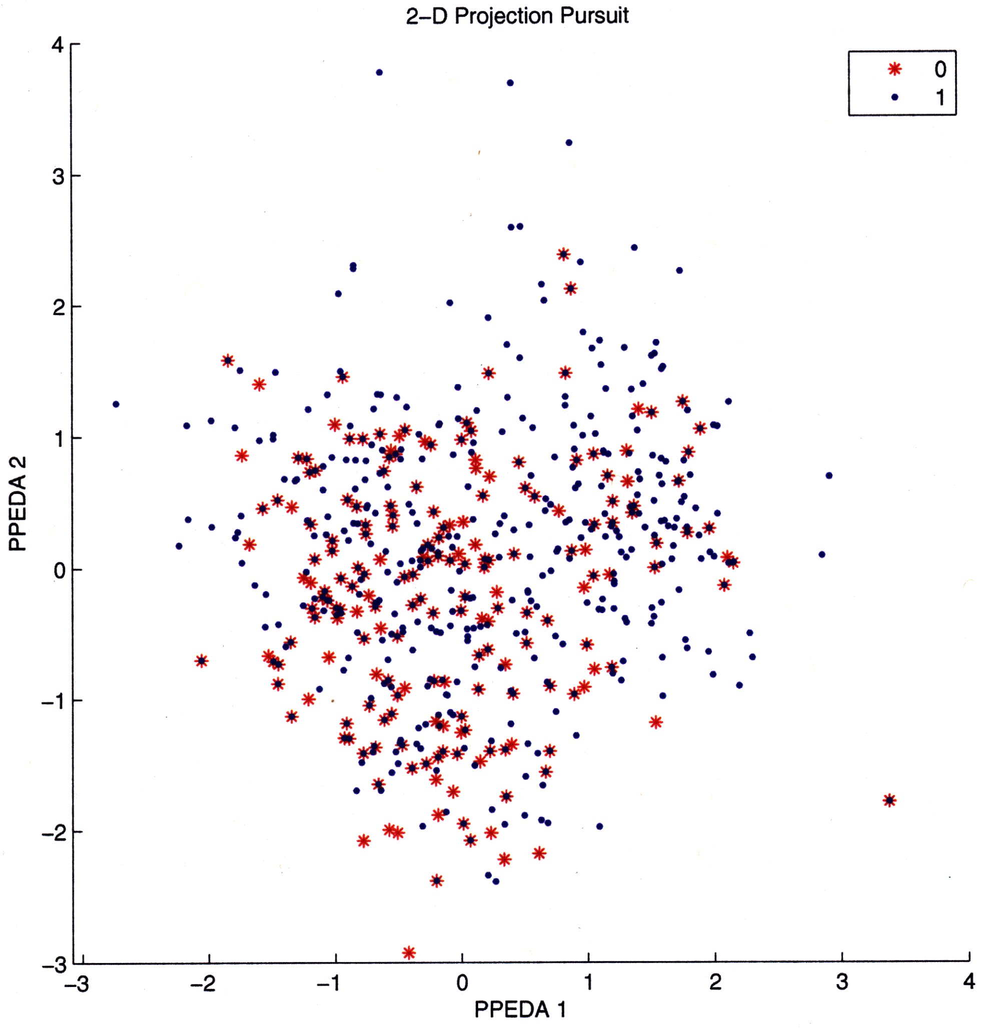


Figure 4-4: Software 2-D Projection Pursuit: There is no separation of clusters through 30 trial runs

Table 4.3: Linear SVM performance

	Correct	Incorrect	Total
Rate	0.5524	0.4476	1
Amount	2762	2238	5000

Results reveal no separation of the histograms.

4.3.4 Support Vector Machines

Given that the other methods have failed to produce patterns or cluster separation, support vector machines provide a final detailed look into possible non-linear separations. Below we explore a variety of non-linear kernels in an attempt to find a non-standard separation of the clusters.

Note that half of the data is chosen at random to train and the other half is used to test the classification. If a high level of correct classification is achieved, we may assume that the SVM decision boundary has found an optimal separation of the two clusters.

Linear Kernel

This is the basic linear separator. The SVM algorithm tries to separate the clusters via an 11-d hyperplane. The results confirm that no linear separation of the data is possible (as was observed in previous methods).

Polynomial Kernel

This is the polynomial kernel separator. The SVM algorithm tries to separate the clusters via an 11-d hyperplane in the transformed polynomial Kernel Space given by:

$$K(\mathbf{w}, \mathbf{x}) = (\mathbf{w} \cdot \mathbf{x} + 1)^n$$

While it is possible to try many orders, testing reveals all orders to show similar performance. The results for polynomial of order 3 shown below.

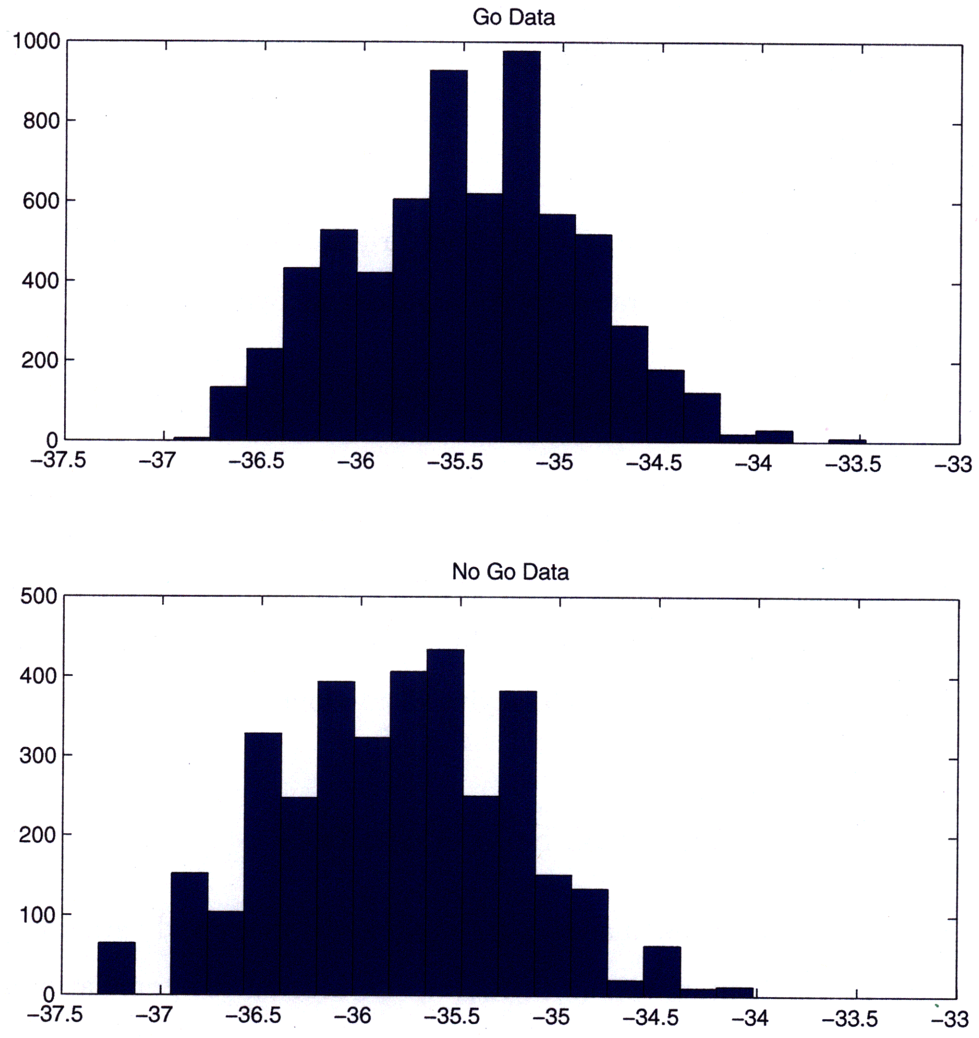


Figure 4-5: 1-D Linear Projection for cluster separation: There is no separation of clusters as seen in the histograms

Table 4.4: Polynomial order SVM performance

	Correct	Incorrect	Total
Rate	0.5166	0.4834	1
Amount	2583	2417	5000

Table 4.5: Radial Basis SVM performance

	Correct	Incorrect	Total
Rate	0.5472	0.4528	1
Amount	2736	2264	5000

Radial Basis Kernel

This is the radial basis separator. The SVM algorithm tries to separate the clusters via an 11-d hyperplane in the transformed radial Kernel Space given by:

$$K(\mathbf{w},\mathbf{x}) = e^{\frac{-\|\mathbf{w}-\mathbf{x}\|^2}{2\sigma^2}}$$

This kernel is particularly useful in radially separated data.

We conclude from the many trials performed that SVM methods fail to identify a linear or non-linear pattern in the data. In all cases the test data is classified with around 50% accuracy, very close to classifying at random.

4.3.5 Mean and Variance Space Analysis

One final method used in the analysis is to analyze the mean and variance space of L,a,b values. The motivation behind this takes into account the granularity issue: If we are to make sense of output data that originates from the same input data, we may group the output and represent the group with two identifying quantities, the mean and variance. In this way, every input will have one group of 6 variables (the mean and variance of each of L,a,b) rather than a set of 6 to 8 vectors of 3 dimensions. Moreover, this 6 variable group will be a consistent representitive of the group, which is an improvement over the generalization technique used previously for Go, No-go clusters.

The handicap of this technique however, is that it prevents us from using binary clustering and decision boundaries. To tackle this, we utilize a 2-means clustering technique. The task then is to identify the clusters that have the greatest separation of means and the smallest variance (i.e. distant and tight clusters). In this way we

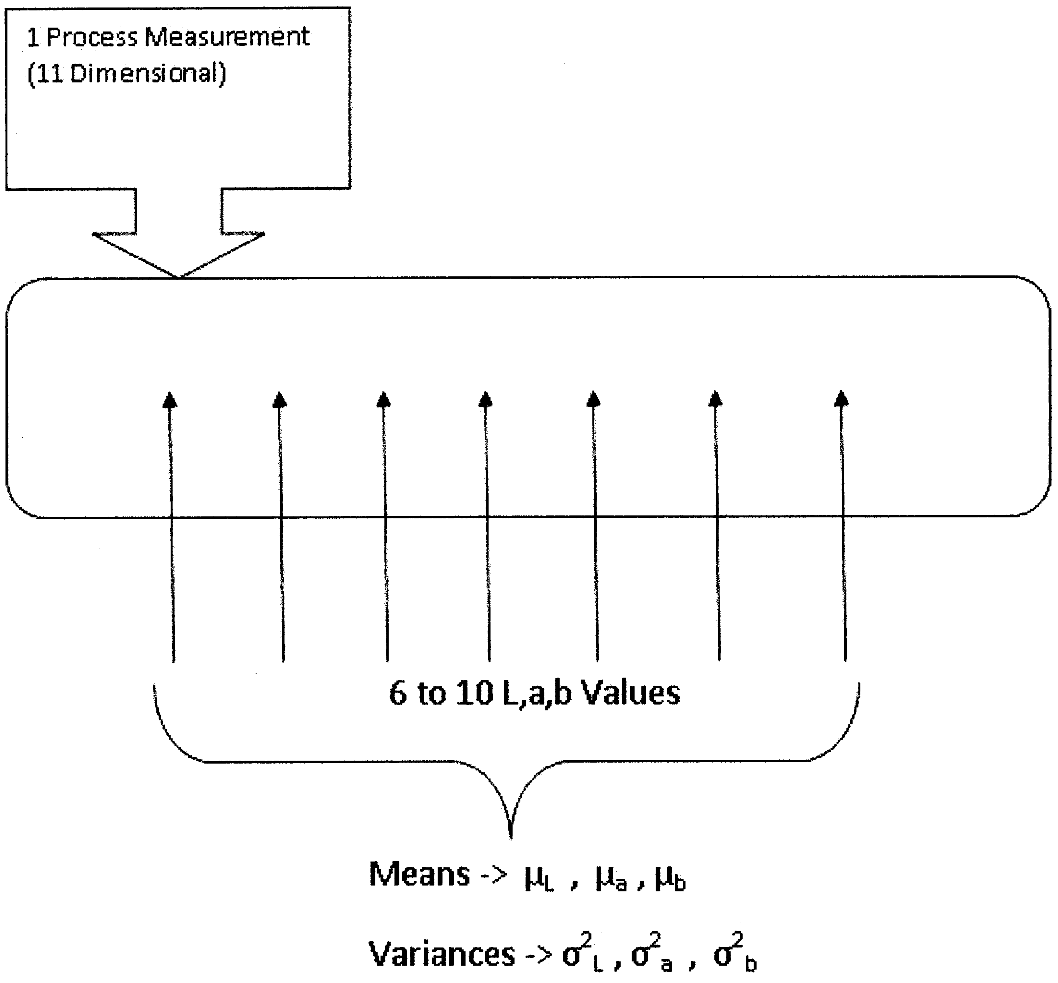
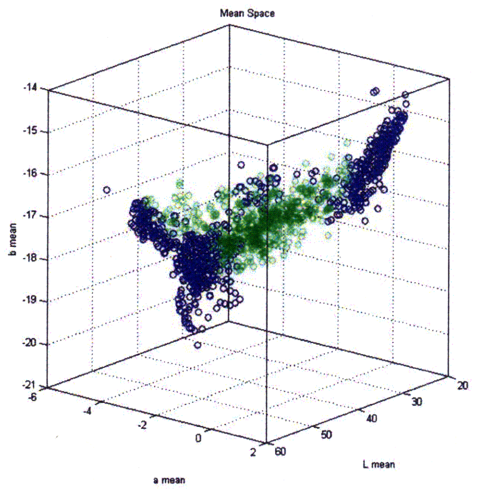
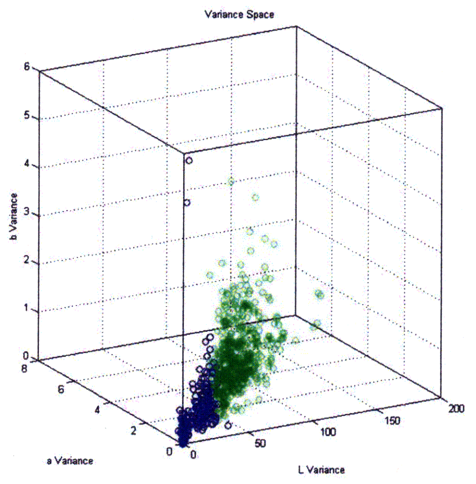

1 Process Measurement
(11 Dimensional)
6 to 10 L,a,b Values
Means -> μ_L , μ_a , μ_b
Variances -> σ^2_L , σ^2_a , σ^2_b

Figure 4-6: Each process measurement corresponds to 6-10 continuous L,a,b measurements



(a) 2-Means clusters on L,a,b mean space



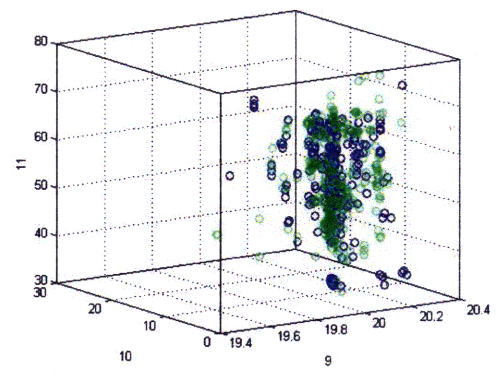
(b) 2-Means clusters on L,a,b variance space

Figure 4-7: 2-Means Clustering on mean and variance data

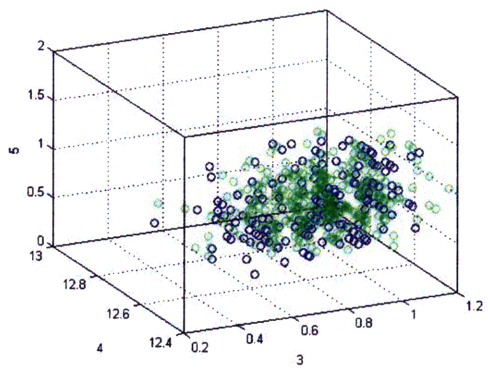
will have obtained the maximum separation of in spec and out of spec groups, which we may then associate with the inputs that lead to them. 2-means clustering in this manner leads to the figures below.

The two figures are the 3-D mean space and variance space. As seen, the clusters are separated as blue and green. Assuming that the blue cluster is for points within spec and green for points out of spec the figures can be explained as follows: The blue cluster is separated into two groups. This separation is owing to the fact that there are 2 washing prescriptions that yield in 2 areas in which the L,a,b values can be within spec. This is because the different washes yield in different L,a,b value ranges that are within specifications for that type of wash. The green cluster is everything outside of these two within spec regions. As can be seen in the variance space, the blue cluster has minimal variance and green has high variance, as may be expected of within spec and out of spec clusters respectively.

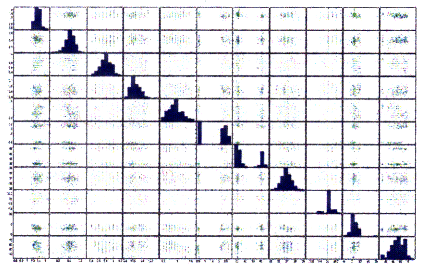
The 2-means clustering technique works well to provide us with a logical pattern. The next step, however, is to associate the findings in mean space and variance space with the actual 11 dimensional measurement vectors. This will allow us to identify



(a) 3-D Plot of Indigo Conc. Hydro Conc and Caustic Flow



(b) 3-D Plot of Indigo temp, Speed and Humidity



(c) 2-D plots of all variable combinations

Figure 4-8: Plots of the cluster correspondance between mean-variance space clusters and the 11-D measurement vectors

which measurements correspond to the in spec cluster and which are linked to the out of spec clusters. We therefore associate the blue and green clusters with the 11-D measurement points that they relate to. This association does not result in an identifiable pattern. As seen in the figures below (which are a few sample 3-D plots chosen from the 11 dimensions and a 2-D plot of each 2 variable combination) the distribution of the clusters in the 11-D vector measurement space is not correlated with the results in mean and variance space.

This random distribution is likely due to the measurement points being snapshots in time that are distanced by an hour. In order to come up with meaningful results, more frequent and representitive measurements are required.

4.4 Developing Decision Strategies

As our data has provided little information and the methods have not revealed a sound classification or decision boundary, we will not be able to provide a decision strategy for the firm. Instead, however, this thesis explores a possible implementation of decision strategy in the form of visual software.

4.4.1 Motivation

While it would be possible to supply businesses with raw data results and mathematical decision boundaries to base their preemptive quality control on, we believe that this is not desirable for many reasons. First, process controllers are often under time constraints and lack the mathematical expertise to decode the decision boundary and make decisions. Second, as our aim is to facilitate and improve production processes, it would be advantageous to provide easy to use decision tools. Finally, we would want a tool that is generic and modifiable as decision strategies change, to continue to function as a decision maker.

4.4.2 Approach

If we are to provide factories with easy to implement and use decision making software, we must leverage the power of software with the strengths of humans. Namely, we will once again utilize the human pattern recognition ability in order to facilitate easy decision making. However, because our decision boundaries will often occur in high-dimensional space, it would not be possible to use traditional graphs in making process decisions.

Chernoff faces provide a very intuitive method to visualize higher dimensions. As humans are very good at recognizing faces and emotions, representing each variable with a facial feature will allow very quick recognition of the state of the process.

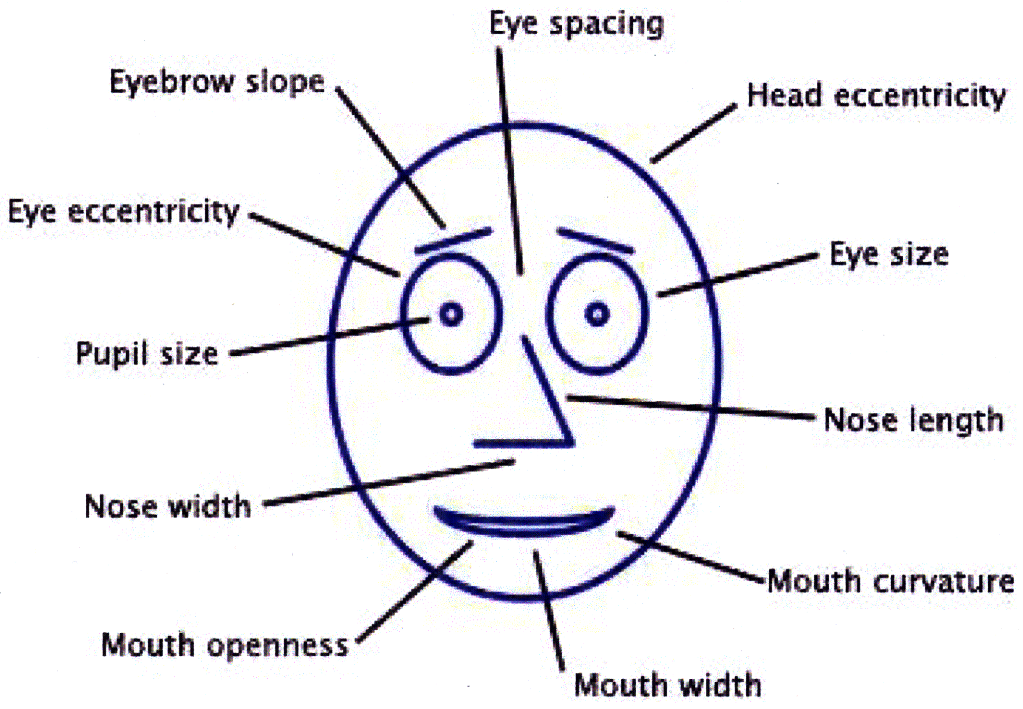


Figure 4-9: Chernoff Face: Each variable corresponds to one of 11 features on the face

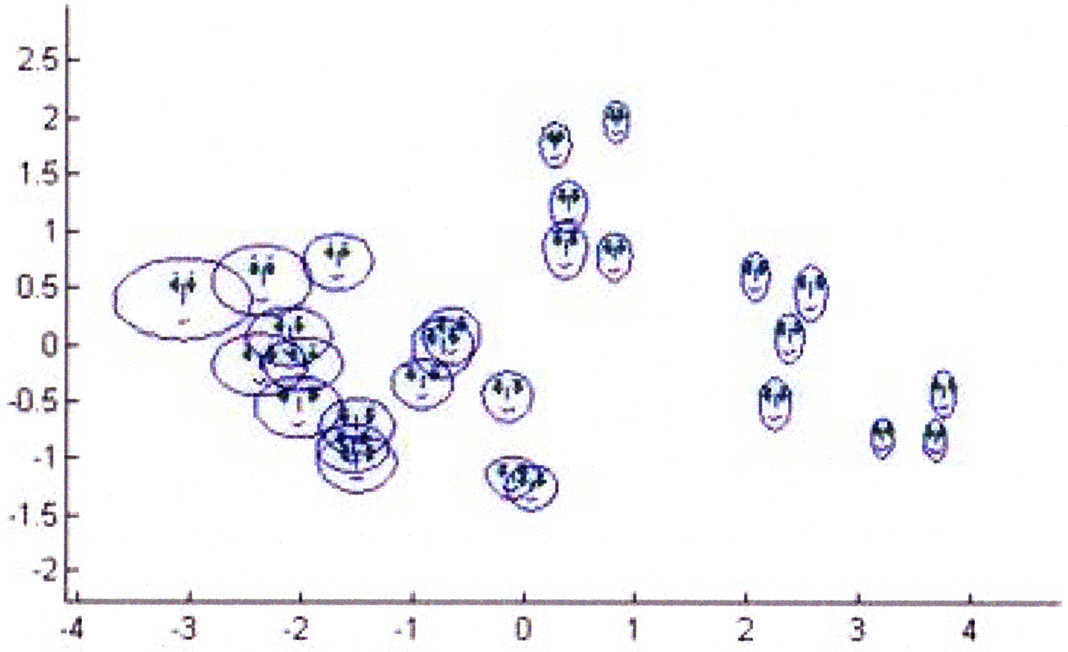


Figure 4-10: Each face may be plotted on a 2-d graph to identify historical trends.

For example, let us explore three random variables -temperature, acidity and pressure (T, a, p) - for a given process. Let us assume that each of the three variables is undesirable at higher levels. Then let us make the following assignments:

- Slope of eyebrows $\propto T$
- Upcurve of mouth arc $\propto a$
- Separation of lips $\propto \frac{1}{p}$

In this case we see that low values of all will yield in a laughing face whereas high values of all will yield in a frowning face. More important than these two end states, however, is the partial states.

For example, if only temperature increases the face will look mischievous, while if only pressure increases the laugh will transform into a contented smile. Therefore, each change gives a clear picture of a different state as well as a gradual change from laugh to frown. These changes would allow quick interpretation by controllers of the condition of the process.

4.4.3 Method

We develop the Chernoff Face software in Java. The software involves a few components.

- Data input and output
- Modeling the face and associating variables
- Establishing decision boundary and face states

Data Input and Output

The software is required to take in process data and use it to determine the state of the face. We use comma deliminated text format in order to take in the input process parameters. The software accepts updates to the document as it is modified in real time by the process measuring instruments. The updates are taken into account at time intervals that are synced with data recording times.

Output data includes both a visual representation of the current state as well as the states of the face drawn on 2-D axes that show both the current length of fabric and time stamp. (Note that these axes are industry specific).

Modeling the face and variable association

Our Chernoff Face Model takes up to 10 parameters of facial features and 1 parameter for color. The features are:

- Head Eccentricity
- Eye eccentricity
- Pupil Size
- Eyebrow Slope
- Nose Location
- Mouth Curl
- Eye Spacing
- Eye Size
- Mouth Size
- Mouth opening

Each feature may take a parameter from the inputs. However, given that decision boundaries may have more complicated mathematical equations, the features may be associated with a function of the inputs rather than the inputs themselves.

The features change on a scale from 0 to 1, therefore the variable must be transformed onto this scale. This transformation is done in conjunction with the decision boundary and will be explained below. Establishing decision boundary and face states

Given our variable associations, our facial features will change on a scale from 0 to 1. The most generic form of the software will set the features to be 0.5 on the decision boundary, and depending on the emotions we wish to reflect, will tend towards 0 or 1 as we get further away from the decision boundary in either direction.

The most generic form of face states will reflect a smiley face when within specifications and a frown when outside. Thus, when inside the decision boundary for in-specification producing inputs the face will smile (usually implying features close to 0) and outside the decision boundary will yield a frown (close to 1).

The face software is open to modification and customization to change features, establish decision boundaries and emotions to reflect various states of the process.

4.4.4 Generic Example

Let us explore the case with the three assignments to eyebrow slope, mouth openness and curl. As described, the faces will transition between states depending on the level of the variables. The figure below shows the transitions for single variables and all three variables.

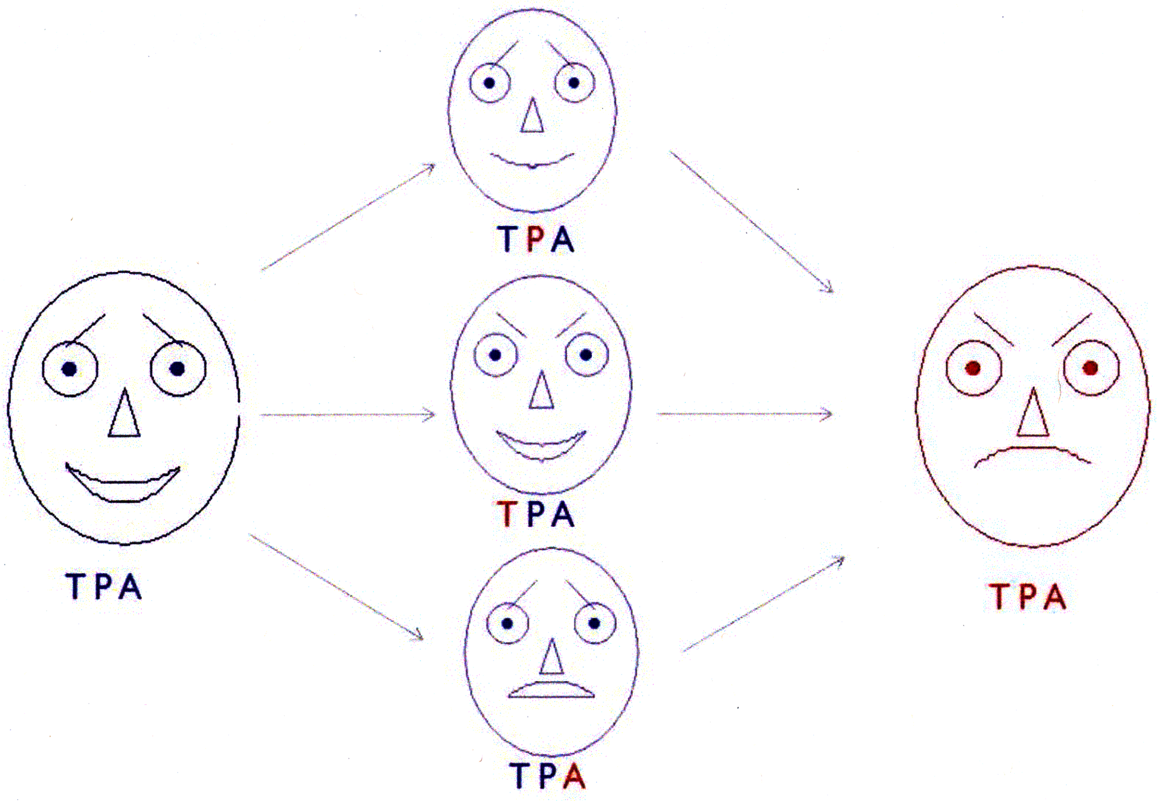


Figure 4-11: The state transitions for the faces: As single variables change in value, the face takes on different emotions. With all variables changing, the face is visibly upset

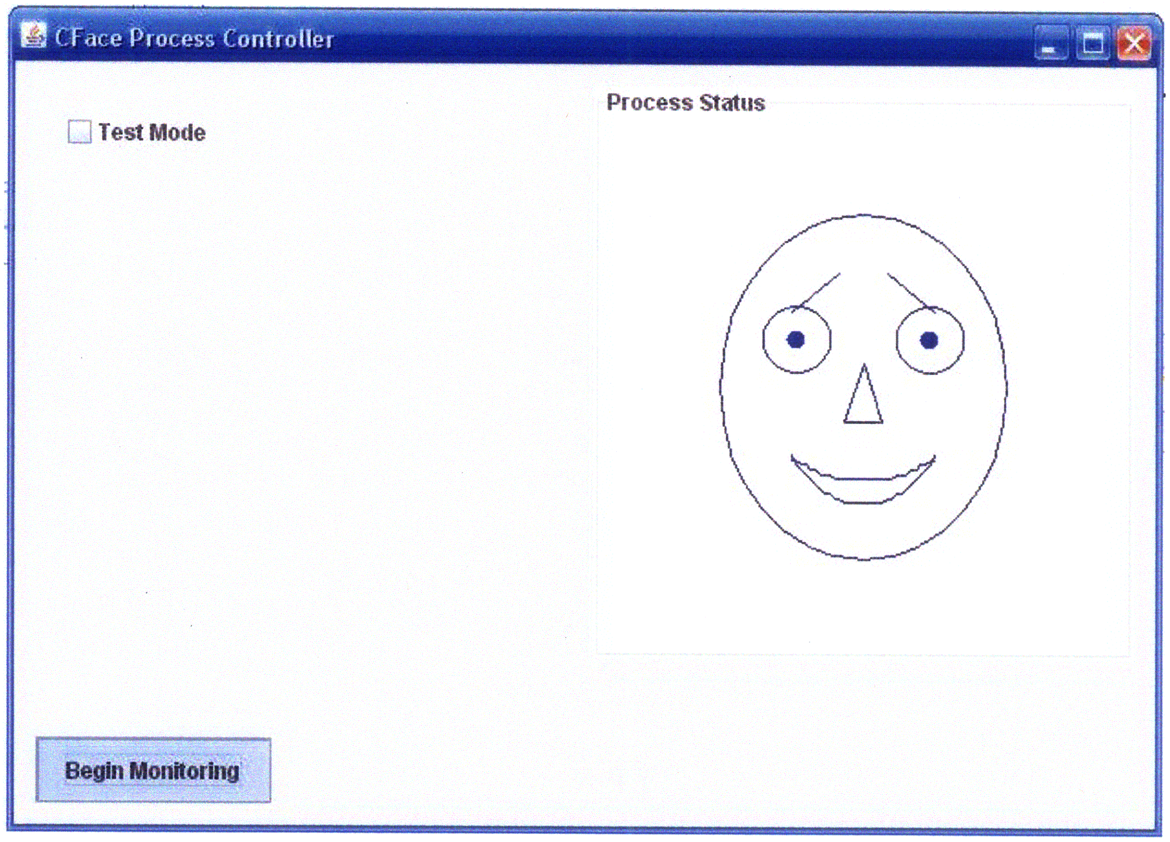


Figure 4-12: Chernoff Face monitoring program: This demonstration program observes data points and visualizes them in Chernoff Faces

Chapter 5

Conclusion

5.1 Results

As presented, our results from looking at Company X data showed clearly that the data did not allow for the proper analysis of inputs and outputs. We explored various reasons for this that may be summarized as follows:

Granularity Many L,a,b values corresponed to few process parameter sets. In fact, there were very few process measurement recordings and these recordings were too scarce (around 10 per work order). Under these conditions it is unlikely to derive meaningful relationships between inputs and outputs.

Redundancy Both L,a,b values and process parameters have little variance. Due to this, there is a lack of unique data points in the data set. Roughly 1 in 10 data points provided are actually unique. Given that we are working in higher dimensional space, and have to deal with the granularity issue, redundancy in data makes it even more scarce and difficult to identify patterns.

Confounding Variables While non-linear dependancy might be the reason for the lack of good regression results, it is also likely that there are significant confounding variables that are not recorded and, thus, cannot be considered. Examples of this might be the type and quality of the cotton mix in the yarn, or various ambient condition variables.

Problems Linking the data Finally, it must also be noted that the linking done between inputs and outputs have a significant drawback. While the input data is linked to certain length values through our extrapolation technique, these length values are in *yarn length* whereas the L,a,b values correspond to lengths in *fabric length.* Because of this, a direct association between the two length values could not be made. It may be thought to be approximately correct due to granularity issues. This is because due to the granularity issue, one process point is likely to correspond to large batches of yarn, which is likely to translate entirely into a batch of fabric such that a one-to-one correspondance between the two may be made. However, this approximation is not a reliable one and we would like to have a method of associating the two length values, which is not possible under current data recording in the factory.

5.1.1 Suggestions for Company X

From the summary of issues presented above, some of them can be prevented or improved upon, while others are inherent to the system. As far as improvements we suggest enhancements to the data collection process.

First, it is important that the granularity issue be solved. It is understood that the granularity issue arises from and is linked to the lack of homogeneity in the tanks and process parameters throughout the process. The solutions to this are twofold: improvement in measurement methods and improvement in dyeing technology. During visits with Company X, it was observed that the factory works closely with machine design firms to improve upon current technology. In this sense, it may be possible to improve the machines to introduce homogenizing techniques. This would aid both in rapid response making the technology more efficient as well as allowing for the accumulation of more data points. Given that this is a more complicated progress, perhaps more achievable is an improvement in measurement. Measurement points may be made to coincide with the exact locations that yarn is dyed. In this way, even if the tank is not homogenous, we may be able to identify the exact conditions under which the yarn is exposed to the dye. This will give us more confidence in obtaining more frequent measures. Another way to accomplish this would be to add more sensors in the tanks.

A final suggestion in obtaining frequent measures is the discovery of the time constant with which the tank propagates the parameters and homogenizes. If we can model this spread, then it will be possible to extrapolate true parameter values from the measurements, even though they do not represent the state of the entire tanks. This discovery may be obtained through further research into the dyeing process. Having more frequent measurements (despite the inherent error) would also aid in this research as well as ours.

Secondary to the granularity issue is a way to link quality control data with various points in process. It is evident from our study that, while records of data are kept dilligently within each process, there is very little tracking done across processes. There are many factors that confound the tracking process. For example, while it is possible to extrapolate length data for the yarn, this does not correlate directly with lengths of fabric post-weaving. Tracking methods need to be implemented to solve this problem so that a given unit may be tracked throughout the process - from the point at which the cotton is spun to the quality control station. One way to accomplish this would be to identify a small unit (such as a certain length) and assign it serial numbers. At each processing point, the effects of the process must be thought out and an association between incoming serial numbers and outgoing serial numbers must be made.

Third, the process parameters are likely to be insufficient in explaining L,a,b variability. This may be solved by careful thought as to what other pre-dyeing factors may be affecting the dyeing process. Examples of this may be the quality and type of cotton mixed into the yarn, conditions under which it is spun and variables relating to the ambient conditions throughout the process. This collection of data would be greatly assisted by the second suggestion of having a standardized way to track data.

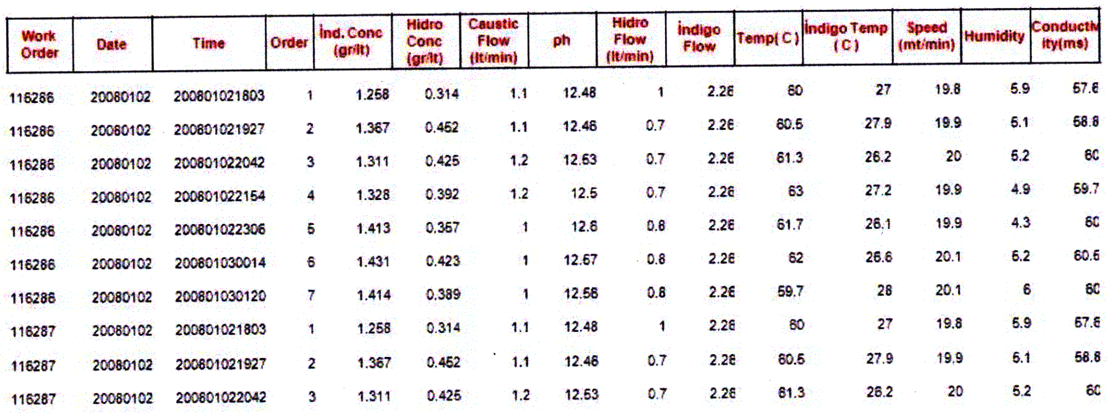
5.2 Contributions

With this thesis I accomplished the following:

- Proposed a methodology and framework to achieve preemptive quality control
- Suggested improvement in Company X data collection that would enable the achievement of preemptive quality control
- Compared various techniques for multivariate analysis of industrial processes with an emphasis on machine learning techniques and visual pattern recognition
- Implemented a decision strategy support vector allowing visualization of high-dimensional state changes

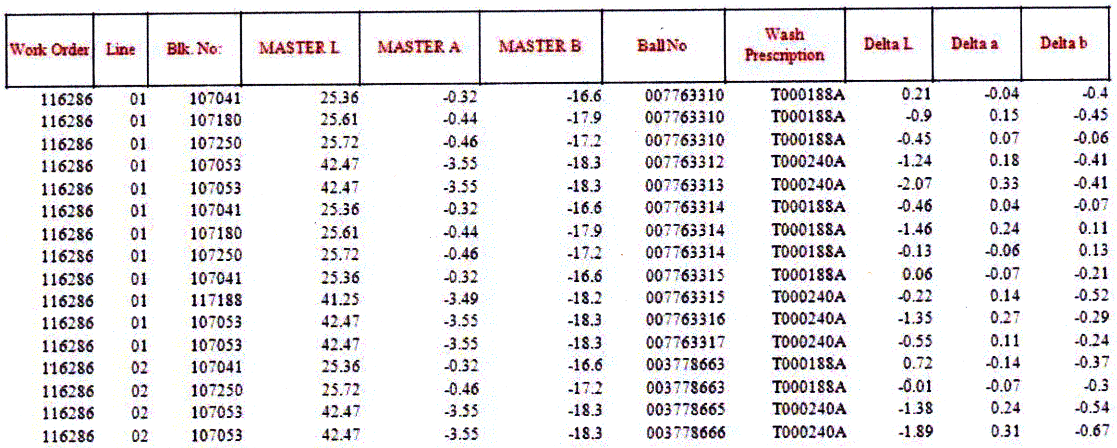
Appendix A

Tables



Work Order	Date	Time	Order	İnd. Conc (gr/lt)	Hidro Conc (gr/lt)	Caustic Flow (lt/min)	ph	Hidro Flow (lt/min)	İndigo Flow	Temp(C)	İndigo Temp (C)	Speed (mt/min)	Humidity	Conductivity(ms)
116286	20080102	200801021803	1	1.258	0.314	1.1	12.48	1	2.26	60	27	19.8	5.9	57.6
116286	20080102	200801021927	2	1.367	0.452	1.1	12.46	0.7	2.26	60.5	27.9	19.9	5.1	58.8
116286	20080102	200801022042	3	1.311	0.425	1.2	12.63	0.7	2.26	61.3	26.2	20	5.2	60
116286	20080102	200801022154	4	1.328	0.392	1.2	12.5	0.7	2.26	63	27.2	19.9	4.9	59.7
116286	20080102	200801022306	5	1.413	0.357	1	12.6	0.8	2.26	61.7	26.1	19.9	4.3	60
116286	20080102	200801030014	6	1.431	0.423	1	12.67	0.8	2.26	62	26.6	20.1	5.2	60.5
116286	20080102	200801030120	7	1.414	0.389	1	12.56	0.8	2.26	59.7	28	20.1	6	60
116287	20080102	200801021803	1	1.258	0.314	1.1	12.48	1	2.26	60	27	19.8	5.9	57.6
116287	20080102	200801021927	2	1.367	0.452	1.1	12.46	0.7	2.26	60.5	27.9	19.9	5.1	58.8
116287	20080102	200801022042	3	1.311	0.425	1.2	12.63	0.7	2.26	61.3	26.2	20	5.2	60

Figure A-1: The input process parameters for the indigo dyeing process: There are 11 parameters with associated work order and timestamp



Work Order	Line	Blk. No:	MASTER L	MASTER A	MASTER B	BallNo	Wash Prescription	Delta L	Delta a	Delta b
116286	01	107041	25.36	-0.32	-16.6	007763310	T000188A	0.21	-0.04	-0.4
116286	01	107180	25.61	-0.44	-17.9	007763310	T000188A	-0.9	0.15	-0.45
116286	01	107250	25.72	-0.46	-17.2	007763310	T000188A	-0.45	0.07	-0.06
116286	01	107053	42.47	-3.55	-18.3	007763312	T000240A	-1.24	0.18	-0.41
116286	01	107053	42.47	-3.55	-18.3	007763313	T000240A	-2.07	0.33	-0.41
116286	01	107041	25.36	-0.32	-16.6	007763314	T000188A	-0.46	0.04	-0.07
116286	01	107180	25.61	-0.44	-17.9	007763314	T000188A	-1.46	0.24	0.11
116286	01	107250	25.72	-0.46	-17.2	007763314	T000188A	-0.13	-0.06	0.13
116286	01	107041	25.36	-0.32	-16.6	007763315	T000188A	0.06	-0.07	-0.21
116286	01	117188	41.25	-3.49	-18.2	007763315	T000240A	-0.22	0.14	-0.52
116286	01	107053	42.47	-3.55	-18.3	007763316	T000240A	-1.35	0.27	-0.29
116286	01	107053	42.47	-3.55	-18.3	007763317	T000240A	-0.55	0.11	-0.24
116286	02	107041	25.36	-0.32	-16.6	003778663	T000188A	0.72	-0.14	-0.37
116286	02	107250	25.72	-0.46	-17.2	003778663	T000188A	-0.01	-0.07	-0.3
116286	02	107053	42.47	-3.55	-18.3	003778665	T000240A	-1.38	0.24	-0.54
116286	02	107053	42.47	-3.55	-18.3	003778666	T000240A	-1.89	0.31	-0.67

Figure A-2: The output L,a,b values at color quality control: The L,a,b values have both the master value and the delta from the master value for the given sample as well as associated work order, line no. and ball no.

Appendix B

Code

B.1 Visual Basic for Excel

Below is the Visual Basic code used to link input and output data

```

Sub Generator()

Dim isemri As Long
Dim currentmeter As Long
Dim counter As Integer


isemri = 0

For counter = 0 To Range("statdata").Rows.count
    If Range("statdata").Cells(counter + 1, 1) = isemri Then
    Call processor(counter, currentmeter)
    Else
    isemri = Range("statdata").Cells(counter + 1, 1)
    currentmeter = 0
    Call processor(counter, currentmeter)
    End If
Next counter

End


```

```vba




End Sub

'This subroutine processes the output data and associates them with the input parameters
'associations are done via calculating the ball meters and production speeds

Sub processor(counter As Integer, currentmeter As Long)

Dim i As Integer
Dim j As Integer

Dim isemri As Long
Dim line As Integer
Dim top As Long
Dim meter As Double
Dim blanket As Long
Dim sira As Integer
Dim yikama As String

isemri = Range("statdata").Cells(counter + 1, 1)
line = Range("statdata").Cells(counter + 1, 2)
top = Range("statdata").Cells(counter + 1, 12)
meter = Range("statdata").Cells(counter + 1, 13)
blanket = Range("statdata").Cells(counter + 1, 8)
sira = Range("statdata").Cells(counter + 1, 15)
yikama = Range("statdata").Cells(counter + 1, 14)

If sira = 1 Then currentmeter = currentmeter + meter

For i = 0 To Range("processdata").Rows.count
    If Range("processdata").Cells(i + 1, 1) = isemri Then
    For j = 0 To 20
        If Range("processdata").Cells(i + 1 + j, 1) = isemri Then
            If Range("cumulmeter").Cells(i + 1 + j, 1) >= currentmeter Then
            Range("tocopy").Offset(i + 1 + j, 0).Copy
            Range("topaste").Offset(counter, 0).PasteSpecial (xlPasteValuesAndNumberFormats)
            Exit Sub
            End If
        End If
    Next j
    End If
Next i

End Sub

'this subrouite processes the output data and associates with input parameters via ↩
    calculating the line associations and dividing input parameter rows evenly ↩
    amont each associated line

Sub lineprocess()

Dim i As Integer
Dim j As Integer
Dim amountline As Integer
Dim isemri As Long
Dim currentline As Integer

currentline = 1
amountline = 0
isemri = 0

For i = 0 To Range("statdata").Rows.count
    If Range("statdata").Cells(i + 1, 1) <> isemri Then


    isemri = Range("statdata").Cells(i + 1, 1)
    currentline = 1

    End If

    amountline = Range("linecount").Cells(i + 1, 1)

    Call processinline(isemri, amountline, currentline, i)

    i = amountline + i - 1
    currentline = currentline + 1

Next i


End Sub

Sub processinline(isemri As Long, numofline As Integer, currentline As Integer, ↩
    rowloc As Integer)
Dim i As Integer
Dim j As Integer
Dim k As Integer
Dim totalfrac As Double


Dim starter As Integer
Dim middle As Integer
Dim ender As Integer

starter = 0
ender = 0
middle = 0

Dim startfrac As Double
Dim midfrac As Double
Dim endfrac As Double

startfrac = 0
endfrac = 0
midfrac = 0

Dim startloc As Integer
Dim endloc As Integer

endloc = 0
startloc = 0


totalfrac = 0

    For i = 0 To Range("startline").Rows.count
        If Range("processdata").Cells(i + 1, 1) = isemri Then
        Do Until Range("processdata").Cells(i + 1, 1) <> isemri
            If Range("startline").Cells(i + 1, 1) = currentline And Range("endline"↩
                ).Cells(i + 1, 1) = currentline Then
                middle = middle + 1
                midfrac = midfrac + Range("fracline").Cells(i + 1, 1)
            ElseIf Range("startline").Cells(i + 1, 1) = currentline Then
                starter = starter + 1
                startloc = i
            ElseIf Range("endline").Cells(i + 1, 1) = currentline Then
                ender = ender + 1
                endfrac = endfrac + Range("fracline").Cells(i + 1, 1)
                endloc = i
            End If
            i = i + 1
            Loop
        If (endfrac + midfrac) <= 1 Then
        startfrac = 1 - (endfrac + midfrac)
        Else
        midfrac = 1 - endfrac
        End If
        Exit For
        End If
    Next i

If CInt(numofline * endfrac) + CInt(numofline * startfrac) + CInt(numofline * ↩
    midfrac) < numofline Then
    startfrac = startfrac + 1 / numofline
End If

j = 0

If endfrac > 0 Then
For j = 0 To CInt(numofline * endfrac) - 1
Range("tocopy").Offset(endloc, 0).Copy
Range("topaste2").Offset(rowloc + j, 0).PasteSpecial (xlPasteValues)
Next j
End If

If midfrac > 0 Then
    For k = 0 To middle - 1
        For j = j To ((CInt(numofline * midfrac) / (middle)) + j - 1) + k
        Range("tocopy").Offset(endloc + k, 0).Copy
        Range("topaste2").Offset(rowloc + j, 0).PasteSpecial (xlPasteValues)
        Next j
    Next k
End If

If startfrac > 0 Then
For j = j To CInt(numofline * startfrac) + j - 1
Range("tocopy").Offset(startloc, 0).Copy
Range("topaste2").Offset(rowloc + j, 0).PasteSpecial (xlPasteValues)
Next j
End If

End Sub

Function countline(i As Integer)
Dim j As Integer
Dim theline As Integer
theline = Range("linesof").Cells(i + 1, 1)
j = 0
Do Until theline <> Range("linesof").Cells(j + i + 1, 1)
    j = j + 1
    Loop
countline = j

End Function




'this subroutine is used to delete data that is corrupted or unusable

Sub deletor()

Dim i As Integer

For i = 0 To Range("statdata").Rows.count
    If Range("zeros").Offset(i + 1, 0).Value = 1 Then
    Range("todelete").Offset(i + 1, 0).Delete
    i = 0
    End If
Next i


End Sub

Sub generalize()

Dim i As Integer
Dim count As Integer

Dim time As Double
Dim amount As Integer
Dim j As Integer

count = 0
time = Range("linetiming").Cells(1, 1)
amount = 0

For i = 0 To Range("totalgo").Rows.count
    If time = Range("linetiming").Cells(i + 1, 1) Then
        amount = amount + Range("totalgo").Cells(i + 1, 1)
```

```
            count = count + 1
        Else
            If amount < count Then
                For j = 0 To count - 1
                Range("linegeneral").Cells(i + j + 1 - count, 1) = 0
                Next j
            Else
                For j = 0 To count - 1
                Range("linegeneral").Cells(i + j + 1 - count, 1) = 1
                Next j
            End If

            time = Range("linetiming").Cells(i + 1, 1)
            count = 0
            amount = 0
            i = i - 1
        End If
Next i


End Sub
Sub generalize2()

Dim i As Integer
Dim count As Integer

Dim time As Double
Dim amount As Integer
Dim j As Integer

count = 0
time = Range("linetiming").Cells(1, 1)
amount = 0

For i = 0 To Range("bgo").Rows.count
    If time = Range("linetiming").Cells(i + 1, 1) Then
        amount = amount + Range("bgo").Cells(i + 1, 1)
        count = count + 1
    Else
        If amount < count Then
            For j = 0 To count - 1
            Range("bgen").Cells(i + j + 1 - count, 1) = 0
            Next j
        Else
            For j = 0 To count - 1
```

```
                Range("bgen").Cells(i + j + 1 - count, 1) = 1
                Next j
            End If

            time = Range("linetiming").Cells(i + 1, 1)
            count = 0
            amount = 0
            i = i - 1
        End If
Next i

End Sub

Sub processdelta()

Dim isemri
isemri = 0

For i = 0 To Range("linecount").Rows.count
    If Range("topaste2").Offset(i, 0) = isemri Then
       Range("topaste3").Offset(i, 0).PasteSpecial (xlPasteValuesAndNumberFormats)
    Else
        isemri = Range("topaste2").Offset(i, 0)
        For j = 0 To Range("processcount").Rows.count
            If Range("processref").Offset(j, 0).Cells(1, 1) = isemri Then
               Range("processref").Offset(j, 0).Copy
               Exit For
            End If
        Next j
        Range("topaste3").Offset(i, 0).PasteSpecial (xlPasteValuesAndNumberFormats)
    End If
Next i




End Sub

Sub seperate()

current = 0

For i = 0 To Range("sepdata").Rows.count
    If Range("sepdata").Cells(i + 1, 4) = 1 Then
    Else
    Range("sepcopy").Offset(i, 0).Copy
    Range("start").Offset(current, 0).PasteSpecial (xlPasteValuesAndNumberFormats)
    current = current + 1
    End If
    Next i


End Sub


Function getmeter(isemri)

For i = 0 To Range(isemri).Rows.count
    If Range(isemri).Cells(i + 1, 1) = isemri Then
    getmeter = Range(linemeters).Cells(i + 1, 1)
    Exit For
    End If
Next i

End Function

'This subroutine is used to associate process parameters with the production lines ↩
    they are associated to
'It outputs the associated start and end line of each line of parameters as well as↩
     the amount in fractions of the end line completed under these parameters

Sub lineassoc()

Dim i As Integer
Dim lineno As Integer
Dim toprocess As Long
lineno = 1
toprocess = 1000

For i = 0 To Range("processdata").Rows.count
    If Range("processdata").Cells(i + 1, 4) = 1 Then
    lineno = 1
    toprocess = Range("perline").Cells(i + 1, 1)
    Range("startline").Cells(i + 1, 1) = lineno
    End If

    Range("startline").Cells(i + 1, 1) = lineno
    toprocess = toprocess − Range("processmeters").Cells(i + 1, 1)

    If toprocess <= 0 Then
    lineno = lineno + Int(Abs(toprocess) / Range("processmeters").Cells(i + 1, 1))↩
        + 1
    Range("fracline").Cells(i + 1, 1) = (Abs(toprocess) Mod Range("processmeters").↩
        Cells(i + 1, 1)) / Range("perline").Cells(i + 1, 1)
    toprocess = Range("perline").Cells(i + 1, 1) + (toprocess Mod Range("↩
        processmeters").Cells(i + 1, 1))
    End If

    Range("endline").Cells(i + 1, 1) = lineno

Next i
End Sub
```

B.2 Chernoff Faces Software in Java

Below is the Chernoff Face application code that relies on the face drawing method provided by John Wiseman [7] Implements Swing GUI.

```






package cfaces;

public class Cfaces extends javax.swing.JFrame {

    private int mode;
    /** Creates new form CfacesGUI */
    public Cfaces() {
        initComponents();
        mode = 0;
    }

    /** This method is called from within the constructor to
     * initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is
     * always regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
```

```java
    //<editor-fold defaultstate="collapsed" desc="Generated Code">
    private void initComponents() {

        beginMonitor = new javax.swing.JToggleButton();
        borders = new javax.swing.JPanel();
        theFace = new FacePanel();
        setMode = new javax.swing.JCheckBox();

        setTitle("CFace Process Controller");
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

        beginMonitor.setText("Begin Monitoring");
        beginMonitor.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
               try {
                    beginMonitorActionPerformed(evt) ;
               } catch (java.io.IOException e)
               {}
                catch (java.lang.InterruptedException f) {}
            }
        });

        borders.setBackground(new java.awt.Color(255, 255, 255));
        borders.setBorder(javax.swing.BorderFactory.createTitledBorder("Process ↩
            Status"));

        javax.swing.GroupLayout theFaceLayout = new javax.swing.GroupLayout(theFace↩
            );
        theFace.setLayout(theFaceLayout);
        theFaceLayout.setHorizontalGroup(
            theFaceLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.↩
                LEADING)
            .addGap(0, 266, Short.MAX_VALUE)
        );
        theFaceLayout.setVerticalGroup(
            theFaceLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.↩
                LEADING)
            .addGap(0, 262, Short.MAX_VALUE)
        );

        setMode.setText("Test Mode");
        setMode.addItemListener(new java.awt.event.ItemListener() {
            public void itemStateChanged(java.awt.event.ItemEvent evt) {
                setModeItemStateChanged(evt);
            }
        });


        javax.swing.GroupLayout bordersLayout = new javax.swing.GroupLayout(borders↩
            );
        borders.setLayout(bordersLayout);
        bordersLayout.setHorizontalGroup(
            bordersLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.↩
                LEADING)
            .addGroup(bordersLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(theFace, javax.swing.GroupLayout.PREFERRED_SIZE, ↩
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.↩
                    PREFERRED_SIZE)
                .addContainerGap())
        );
        bordersLayout.setVerticalGroup(
            bordersLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.↩
                LEADING)
            .addGroup(bordersLayout.createSequentialGroup()
                .addContainerGap()
                .addComponent(theFace, javax.swing.GroupLayout.PREFERRED_SIZE, ↩
                    javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.↩
                    PREFERRED_SIZE)
                .addContainerGap())
        );


javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.↩
                    Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(beginMonitor))
                    .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.↩
                        createSequentialGroup()
                        .addGap(25, 25, 25)
                        .addComponent(setMode)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement↩
                            .RELATED, 209, Short.MAX_VALUE)
                        .addComponent(borders, javax.swing.GroupLayout.↩
                            PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, ↩
                            javax.swing.GroupLayout.PREFERRED_SIZE)))
                .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.↩
                createSequentialGroup()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.↩
                    Alignment.LEADING)
                    .addGroup(layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(borders, javax.swing.GroupLayout.↩
                            PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE, ↩
                            javax.swing.GroupLayout.PREFERRED_SIZE))
                    .addGroup(layout.createSequentialGroup()
                        .addGap(26, 26, 26)
                        .addComponent(setMode)))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED↩
                    , 44, Short.MAX_VALUE)
                .addComponent(beginMonitor, javax.swing.GroupLayout.PREFERRED_SIZE↩
                    , 36, javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap())
        );
        pack();
    }// </editor-fold>

    private void beginMonitorActionPerformed(java.awt.event.ActionEvent evt) throws↩
         java.io.IOException, java.lang.InterruptedException {

        if (mode == 1) {
        theFace.run();
        }
        else {
        theFace.go();
        }
}
     private void setModeItemStateChanged(java.awt.event.ItemEvent evt) {
        if (evt.getStateChange() == java.awt.event.ItemEvent.DESELECTED)
            mode = 0;
        else if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED)
            mode = 1;
    }

    /**
    * @param args the command line arguments
    */
    public static void main(String args[]) {
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
              Cfaces frame = new Cfaces();
              frame.setVisible(true);
            }
        });
    }

    // Variables declaration - do not modify
    private javax.swing.JToggleButton beginMonitor;
    private javax.swing.JPanel borders;
    private FacePanel theFace;
    private javax.swing.JCheckBox setMode;
    // End of variables declaration

}



package cfaces;

import java.util.Random;


public class Face {

    /**
     * The dimensions of facial characteristics as follows:
     * 1 : head eccentricity
     * 2 : eye eccentricity
     * 3 : pupil size
     * 4 : eyebrow form
     * 5 : nose location
     * 6 : mouth curl
     * 7 : eye spacing
     * 8 : eye size
     * 9 : mouth size
     * 10: mouth opening
     */
    public double dim[] = new double[11];


    public Face() {
        int i;
        Random r = new Random();

        for (i = 1; i < 11; i++) {
            dim[i] = r.nextDouble();
        }
    }

    public Face(String[] dims) {
        int i;
        for (i = 0; i < dims.length;i++) {
            dim[i+1] = Double.parseDouble(dims[i]);
        }
    }

    public Face(double dim1, double dim2, double dim3, double dim4, double dim5, ↩
        double dim6, double dim7, double dim8, double dim9, double dim10) {
        dim[1] = dim1;
    dim[2] = dim2;
    dim[3] = dim3;
    dim[4] = dim4;
    dim[5] = dim5;
        dim[6] = dim6;
    dim[7] = dim7;
    dim[8] = dim8;
    dim[9] = dim9;
    dim[10] = dim10;
    }

        public double distance(Face f) {
                int i;
                double total = 0;
                double dif;

                for (i = 1; i < 11; i++) {
                        dif = dim[i] - f.dim[i];
                        total = total + dif * dif;
                }
                return Math.sqrt(total);
        }

    public int direction(Face f) {
     double dif;
     double dir;
     int i;

     dir = 0;

     for (i = 1; i < 11; i++) {
                         dif = dim[i] - f.dim[i];
            dir = dir + dif;
     }

    if (dir >= 0)
        return 1;
    else
        return -1;
    }

    public double size() {
        int i;
        double tot;
        tot = 0;

        for (i=1; i<11; i++) {
            tot = dim[i] * dim[i];
        }
        return tot;
    }
}

package cfaces;
import javax.swing.JPanel;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Image;
import java.io.IOException;
import java.io.StreamTokenizer;
import java.io.StringBufferInputStream;




class FacePanel extends JPanel implements Transition {

        Graphics offscreenContext;
        Image offscreenImage;



    TFace facev = new TFace();
  TFace fbound = new TFace();


    FaceDraw face = new FaceDraw();


    Color foregroundColor = Color.blue;

  Color badColor = Color.red;


    Color backgroundColor = Color.white;



    int speed = 100;


    int fps = 100;

    public void run() {


            Motion motion = new Motion(this);
    fbound = new TFace(0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5);
            TFace face1, face2;
    int i;

            motion.FPS = fps;
            face1 = new TFace();

            for (i=0; i < 8; i++) {

                    face2 = new TFace();

                    motion.doTransition(face1, face2, ((double) 1.0 / speed));
                    face1 = face2;
            }
    }

  public void go() throws IOException, InterruptedException {

            Motion motion = new Motion(this);
```

```
        Input inputs = new Input();
        fbound = new TFace(0.8, 0.5, 1, 0.5, 0.5, 0.5, 0.5, 1, 1, 0.5);
        int i;
                TFace face1, face2;
        face1 = fbound;
        motion.FPS = fps;
        for (i=0; i < 8; i++) {
       String[] dimensions = inputs.regetInput(inputs.getCurrent());
        face2 = new TFace(dimensions);
        motion.doTransition(face1, face2, ((double) 1.0 / speed));
        face1 = face2;
        inputs.setCurrent(inputs.getCurrent()+ 1);
        Thread.sleep(3000);
        }


    }

    public void change(FaceState state) {
                facev = (TFace) state;
        this.paint(this.getGraphics());

        }


        public void paint(Graphics g) {


        offscreenImage = this.createImage(size().width, size().height);
                offscreenContext = offscreenImage.getGraphics();

                if (facev != null) {
                        if (offscreenContext != null && offscreenImage != null) {

                                offscreenContext.setColor(backgroundColor);
                                offscreenContext.fillRect(0, 0, size().width, size↩
                                    ().height);


                                offscreenContext.setColor(foregroundColor);

                if (facev.distance(fbound) > 0 && facev.direction(fbound) == -1)
                    offscreenContext.setColor(badColor);

                                face.draw(offscreenContext, facev, 0, 0, size().↩
                                        width, size().height);


                                    g.drawImage(offscreenImage, 0, 0, this);
                            } else {

                                    face.draw(g, facev, 0, 0, size().width, size().↩
                                        height);

                    }
        }
    }

        public void update(Graphics g) {
                paint(g);
        }

}



class TFace extends Face implements FaceState {

        public TFace(double p1, double p2, double p3, double p4, double p5, double ↩
            p6, double p7, double p8, double p9, double p10) {
                super(p1, p2, p3, p4, p5, p6, p7, p8, p9, p10);
        }

    public TFace(String[] s) {
        super(s);
    }

        public TFace() {
                super();
        }


        public FaceState translate(FaceState start, FaceState end, double t) {
                int i;
                TFace interpolated = new TFace↩
                    (0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0);

                for (i = 1; i < 11; i++) {
                        interpolated.dim[i] = ((Face) start).dim[i] + (((Face) end)↩
                            .dim[i] - ((Face) start).dim[i]) * t;
                }
                return interpolated;
        }
}





package cfaces;

interface Transition {
    public void change(FaceState state);
}

interface FaceState {
    public FaceState translate(FaceState start, FaceState end, double time);
}

class Motion implements Runnable {
   protected Transition transend;
   protected FaceState start;
   protected FaceState end;
   protected double duration;
   public int FPS = 10;
   protected boolean done = true;

   public Motion(Transition t) {
       this.transend = t;
   }

   public boolean isDone(){
       return done;
   }
   public void doTransition(FaceState start, FaceState end, double duration) {
       this.start = start;
       this.end = end;
       this.duration = duration;
       action();
   }

   public void action() {
       FaceState current;
       int delay = 1000 / FPS;
       double delta = duration;
```

```
        double d;

        for (d = 0.0; d < 1.0; d = d + delta ) {
            current = start.translate(start , end , d);
            transend.change(current);
            try {
                Thread.sleep(delay);
            }
            catch (InterruptedException e){
            }
        }

    }

    public void run(){
        action();
        done = true;
    }
}
```

Bibliography

[1] H. Chernoff. The use of faces to represent points in k-dimensional space graphically. *Journal of the American Statistical Association*, 68(342):361–368, 1973.

[2] E. W. Deming. *Out of the Crisis.* MIT Center for Advanced Engineering Study, Boston, Massachusetts, 1986.

[3] J. A. Hartigan. A k-means clustering algorithm. *Applied statistics*, 28(1):100–108, 1979.

[4] Peter J. Huber. Projection pursuit. *The Annals of Statistics*, 13(2):435–475, 1985.

[5] Klaus-Robert Muller, Sebastian Mika, Gunnar Ratsch, Koji Tsuda, and Bernhard Scholkopf. An introduction to kernel-based learning algorithms. *IEEE Transactions on Neural Networks*, 12(2):181–200, 2001.

[6] K. Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical magazine*, 1901.

[7] John Wiseman. Chernoff faces, July 1998. http://people.cs.uchicago.edu/ wiseman/chernoff/.