SEMICONDUCTOR EQUIPMENT ANALYSIS AND WAFER STATE PREDICTION SYSTEM USING REAL-TIME DATA

by

Sherry Fen-hwei Lee

Memorandum No. UCB/ERL M94/104

15 December 1994

SEMICONDUCTOR EQUIPMENT ANALYSIS AND WAFER STATE PREDICTION SYSTEM USING REAL-TIME DATA

by

Sherry Fen-hwei Lee

Memorandum No. UCB/ERL M94/104

15 December 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

SEMICONDUCTOR EQUIPMENT ANALYSIS AND WAFER STATE PREDICTION SYSTEM USING REAL-TIME DATA

by

Sherry Fen-hwei Lee

Memorandum No. UCB/ERL M94/104

15 December 1994

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

Abstract

Semiconductor Equipment Analysis and Wafer State Prediction System Using Real-Time Data

by

Sherry Fen-hwei Lee

Doctor of Philosophy in Engineering-Electrical Engineering and Computer Sciences

University of California at Berkeley

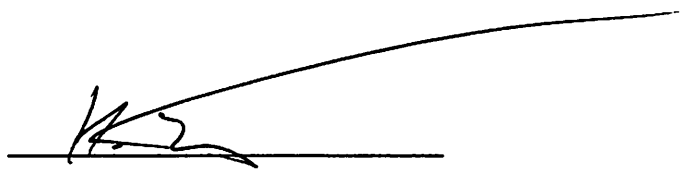
Professor Costas J. Spanos, Chair

The fabrication of modern semiconductor products requires thousands of processing steps. A key element in achieving high yields and throughput with short cycle-times is to monitor the equipment to ensure proper processing at each step. This thesis develops a monitoring method suitable for real-time fault detection, fault diagnosis, and wafer state prediction. Because not all wafer states can be directly measured while the wafers are being processed in each piece of equipment, we use real-time signals sensitive to the equipment state to infer the condition of the wafer. This set of real-time signals is monitored and analyzed by the system, which consists of three distinct modules.

The fault detection module employs time series modeling and multivariate statistics to detect run-time errors on a second-to-second basis. When a malfunction is detected, the fault diagnosis module assigns a cause to the problem. Two methods for diagnosis were investigated. The first uses discriminant analysis techniques, while the second uses a combination of clustering algorithms and neural network models. Examples of faults which have been detected and diagnosed on a plasma etcher include various levels of miscalibrations in mass flow controllers, pressure gauges, and radio frequency (RF) power generators.

In addition, the system predicts the wafer state after each process step. Generally, models for wafer states are built using the input settings of the equipment. Experimental results in this thesis, however, demonstrate that models built with select real-time signals, which we call chamber state based (CSB) models, are effective for the prediction of key wafer states of plasma processes especially after the machine has aged significantly since the original model was created.

The system as a whole has the potential to reduce the overall cost of ownership of semiconductor equipment by increasing both the wafer yield and throughput of product wafers, and decreasing the down-time and mean-time-to-repair of the equipment. Furthermore, this system does not depend upon monitor wafers or expensive metrology; rather, it uses real-time signals collected automatically and non-invasively from the equipment. As such, it will enable inexpensive run-to-run and real-time control applications. The system has been developed and tested on the Lam Rainbow 4400 and Lam TCP 9600 plasma etch equipment.



Committee Chairman

Contents

List of Figures

List of Tables

Acknowledgments

I am grateful to my research advisor, Professor Costas J. Spanos, for his excellent guidance and generous support during the last five years. His vision of the semiconductor manufacturing industry was instrumental in helping to determine the direction of my research. Professor William Oldham, who served as the Chair of my Qualifying Exam Committee and as a member of my M. S. committee, provided insightful advice and suggestions during the writing of this thesis. Professor David R. Brillinger, who served on both my Qualifying Exam Committee and my M. A. Committee in the Statistics Department, has been an excellent source for feedback concerning many statistical aspects of this work. I especially thank Dean David Hodges for his continuous support of the Berkeley Computer Aided Manufacturing (BCAM) group, and for serving on my Qualifying Exam Committee.

Also deserving thanks is Alan Miller of Lam Research for sharing his expertise in plasma etch processes, and for his continued interest in this work. I am grateful to Digital Equipment Corporation for sponsoring me in the form of a fellowship since 1990, and thank my Digital mentors, Dr. Ahsan Enver, Walter Metz, and Mike Coffey for their helpful advice and support.

The bulk of the experiments in this thesis were conducted in the Berkeley Microfabrication Laboratory. I am grateful to the staff, especially to Debra Hebert, Maria Perez, and Dave Hebert for sharing their processing knowledge, and to Bob Hamilton and Evan Stateler for adding the extra hardware to the etcher.

My experience was enhanced, both intellectually and socially, by the valuable interaction with the members of the BCAM group. Special thanks are extended to Antonio Miranda for developing the data collection interface for the Comdel RPM-1 and for devot-

ing countless hours helping to fabricate the experiment wafers. I also appreciate Crid Yu's assistance in developing the layout for the test structure. For their insightful discussions and feedback, I thank the other past and present members of the BCAM group: Eric Boskin, Dr. Norman Chang, Dr. Raymond Chen, Roawen Chen, Sean Cunningham, Mark Hatzilambrou, Christopher Hylands, Herb Huang, Soverong Leang, Dr. Kuang-Kuo Lin, Dr. Zhi-min Ling, H. C. Liu, Lauren Massa-Lochridge, Professor Gary May, David Mudie, Xinhui Niu, David Rodriguez, Manolis Terrovitis, Pamela Tsai, and Haifang Guo Yun. Additional thanks are extended to Tony Miranda and Eric Boskin for proofreading portions of this thesis.

I also thank Daniel Miranda, Henry Sheng, Professor Peter Beerel, Dr. Ken Nishimura, Professor Dawn Tilbury, Lisa Buckman, Sam Sheng, and John O'Brien for their friendship.

This research has been sponsored by Digital Equipment Corporation, the Semiconductor Research Corporation (SRC), SEMATECH (93-MP-700, 94-YP-700), Lam Research, and California Micro with matching funds from National Semiconductor, Texas Instruments, Atmel Corporation, and Advanced Micro Devices.

Chapter 1

Introduction

1.1 Motivation

With the DRAM capacity quadrupling every three years, gigabit chips with linewidths of less than 0.2μm will be in production around the turn of the century [1.1]. To achieve these small linewidths and the resulting high density circuitry, it is predicted that the cost to build a new semiconductor fabrication factory (fab) will exceed the $1 billion mark by 1996 and be in excess of $1.5 to $2 billion in the year 2000 [1.2][1.3]. Over 75% of this capital for a new factory is attributed to equipment cost. Despite the high cost of modern semiconductor equipment, the equipment utilization for product is low, estimated between 35% and 50% [1.2][1.3]. Equipment utilization is defined as the percentage of time that the machine is used to produce good production wafers. As depicted in Figure 1.1, equipment loss, including down-time and time for maintenance, calibrations, set-up, and pilot runs, presently accounts for 28% of equipment time. Another 11% is due from operating loss, which includes the time the equipment misprocesses wafers, waits for material, and

processes bad material. Non-equipment loss, including the time the machine is idle or is used for training, and special work, including the time to make process adjustments, make up another 26%. It is estimated that to remain competitive and profitable in the future, at least 70% to 80% equipment utilization is necessary [1.2][1.3], which will require a significant decrease in the areas of both equipment and operating loss.

A key element in achieving this goal is to monitor the equipment to ensure that the semiconductor wafers are processed properly at each step. The cost in dollars and time to measure each wafer after it completes each step, however, becomes prohibitive in modern semiconductor factories, which produce wafers with well over 100 manufacturing steps. Present practice is to measure monitor wafers periodically, perhaps at the start of each work shift, after performing maintenance, or after changing the machine settings. Even with the use of monitor wafers, however, subsequent production wafers may still be processed improperly.

Currently, final test is generally performed after all the processing steps have been completed, as illustrated in Figure 1.2(a). Thus, instead of detecting equipment faults causing wafer yield loss early in the process flow, wafer yield loss is usually found very late in the processing line. Defective wafers, or scrap, can be extremely costly depending on how many processing steps the wafers have completed. The late detection also makes diagnosis of the problem very difficult. Present practice may require first stripping the problem wafers layer by layer until the fault is isolated, then tracing the fault back to a specific piece or group of equipment. While this technique has enjoyed some success among various yield groups in modern fabs, a more direct approach, catching faults immediately after they have occurred, is much more appealing.

By pushing fault detection earlier in the processing line (Figure 1.2(b)), considerable resources are saved because once a fault has been detected in a particular machine, that

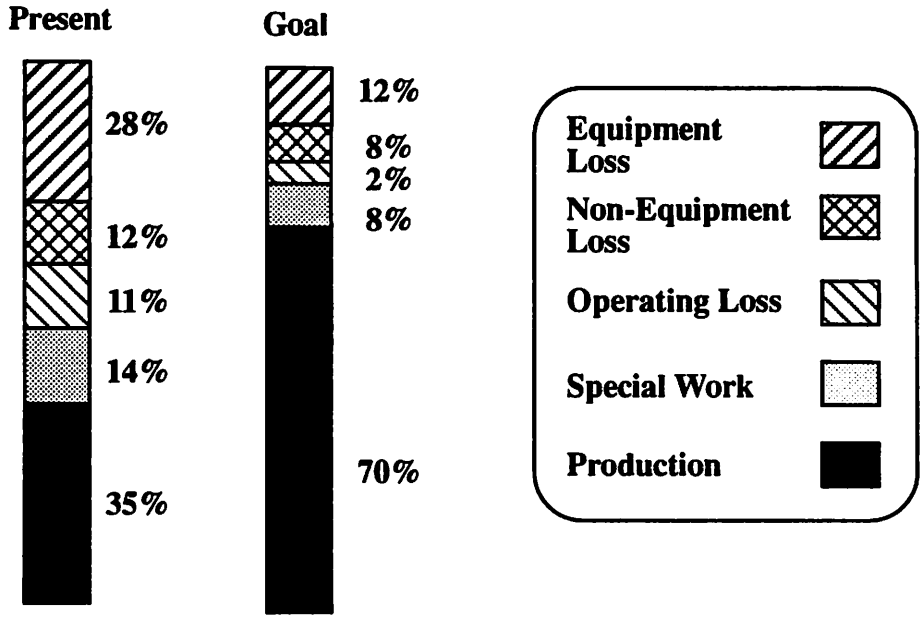


Figure 1.1 Equipment Utilization. To remain profitable and competetive in the future, factories must achieve at least 70% utilization of their equipment [1.2].

machine can be stopped immediately, reducing the number of misprocessed wafers. Because the equipment causing the fault is easily isolated, faster diagnosis of the problem is possible. In addition to detecting and diagnosing problems with the machine, the effect of the fault on the wafer can be assessed, making it possible to ensure that only wafers worth processing continue down the manufacturing line.

1.2 Thesis Overview

This thesis develops a system, called the Equipment Analysis and Wafer State Prediction System, to perform real-time semiconductor equipment fault analysis and prediction of wafer state. The system uses real-time signals automatically collected from the equipment via various real-time monitors. As depicted in Figure 1.3, the real-time data is fed into each of three modules: (1) Fault Detection, (2) Fault Diagnosis, and (3) Wafer State Prediction. Examples of faults detected and diagnosed on a plasma etcher include a faulty

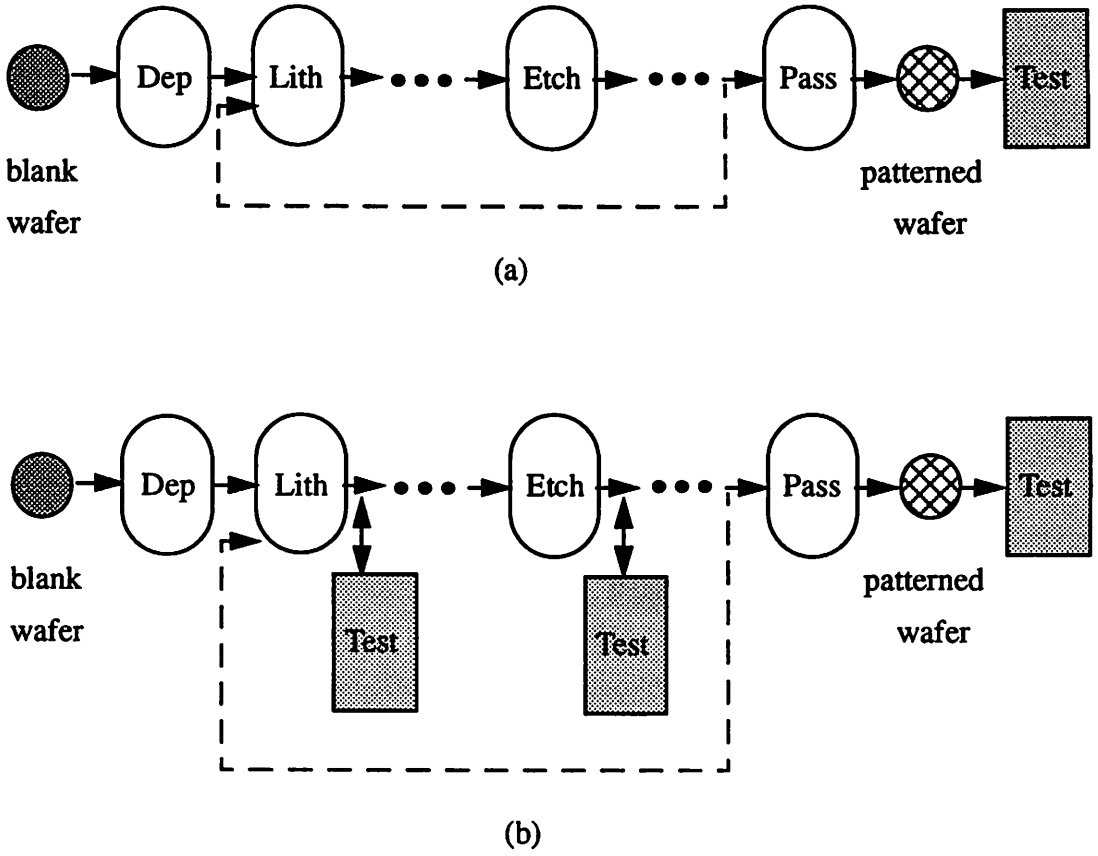


Figure 1.2 (a) Typical Manufacturing Process: Final test is usually done after all the processing steps have been completed. (b) Proposed Manufacturing Process: This thesis pushes fault detection earlier in the manufacturing line, during the processing of each piece of equipment or workcell, allowing for less scrap and faster diagnosis. Additionally, the quality of the wafer can be assessed before it continues down the manufacturing line.

mass flow controller, an unstable power supply, changes in chamber pressure, and a miscalibrated electrode gap spacing. Wafer states of interest may include the etch rate, selectivity, or uniformity of a wafer after it is processed by each piece of equipment. The system as a whole has the potential to reduce the overall cost of ownership of semiconduc-

tor equipment by increasing both the wafer yield and throughput of product wafers, and decreasing the down-time and mean-time-to-repair (MTTR) of the equipment.

Though general enough to be applied to many pieces of semiconductor equipment, the methodology is verified on plasma processing, one of the costliest operations in the semiconductor fabrication line. Plasma processing is not only very expensive, but also is difficult to control because it is not well understood. In fact, a malfunctioning plasma etcher can generate up to $100,000 worth of scrap per hour [1.4]. Although there is a tremendous push to develop models relating the plasma to interesting output characteristics of the wafer based on basic physical principles, researchers are still years away from developing models realistic enough to be useful on the factory floor [1.5][1.6][1.7]. Thus, at this time empirical models are faster and more practical for prediction.

In this work, the models used in each module are empirically based on real-time data collected while the machine is processing wafers. The following sections briefly describe the purpose of each system module. Potential impact areas of the Equipment Analysis and Wafer State Prediction System on the equipment ownership cost are then highlighted.

1.2.1 Fault Detection Module

The Fault Detection Module uses automatically collected real-time data to determine the health of the semiconductor equipment while the wafer is being processed. Two types of faults are determined by the module; the first group of faults corresponds to fast equipment fluctuations within the processing time of one wafer, while the second group reflects longer duration changes in the overall equipment state. The machine status is displayed in a control chart which can be easily read and interpreted by an operator on the factory floor. In this way, the complex modeling algorithms are transparent to the user.

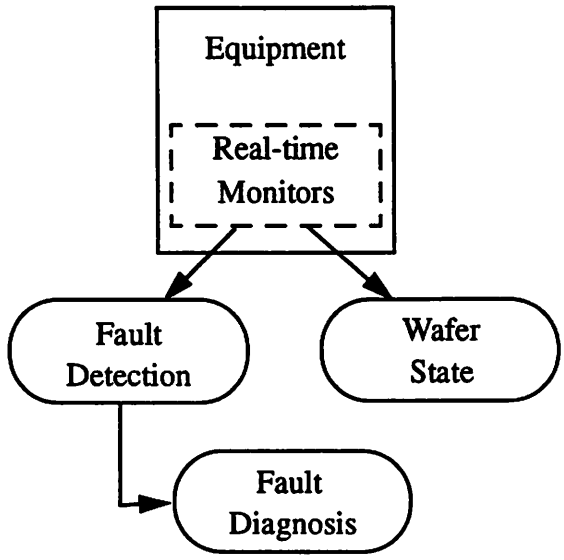


Figure 1.3 Schematic of Equipment Analysis and Wafer State Prediction System: The system contains three modules: (1) Fault Detection, (2) Fault Diagnosis, and (3) Wafer State Prediction. The modules use data collected from real-time monitors.

1.2.2 Fault Diagnosis Module

Once an equipment fault has been detected by the Fault Detection Module, the Diagnosis Module assigns a cause to the problem. In addition to diagnosing faults that have already occurred, this module may also predict impending malfunctions, or perform prognosis of faults. For example, it may be possible to determine when preventive maintenance is needed. Two methods are developed which tackle the difficult task of equipment fault diagnosis and prognosis. The first uses discriminant analysis, while the second uses a combination of clustering techniques and neural network models. The main idea is to map the signature of the real-time signals to a specific equipment fault. Each method has advantages and disadvantages, depending on the type of equipment faults. Examples of

faults that have been diagnosed include faulty mass flow controllers, miscalibrated power supplies, changes in chamber pressure, and a change in the electrode gap spacing.

1.2.3 Wafer State Prediction Module

In addition to detecting and diagnosing equipment faults, it is important to assess the quality of the wafers immediately after each process step. For example, it is useful to know how a particular equipment fault has impacted the processing of the wafer. The Wafer State Prediction module performs this task. Good quality wafers can continue down the fabrication line, while misprocessed wafers can be discarded or reworked. This thesis shows that models based on real-time equipment data result in effective prediction capability for plasma etch processes.

1.3 System Impact on Cost of Ownership

The overall goal of the fab is to obtain high yield with a low cycle-time and high throughput. When implemented on a high volume production line, the Equipment Analysis and Wafer State Prediction System as a whole addresses these issues and can potentially lower the overall ownership cost of the equipment.

A summary of these cost of ownership advantages are listed in Table 1.1. The general categories for this list are adapted from the SEMATECH Cost of Ownership model [1.8]. The fault detection algorithm can reduce the process scrap yield produced by the equipment, defined as the operational yield of the equipment. For etchers in particular, the savings can be considerable, as wafers close to completion can be worth several thousands of dollars.

The diagnostic capability can reduce the equipment down-time. Down-time includes repair time, non-production time during scheduled maintenance, and engineering usage. The repair time is impacted by the mean-time-to-repair (MTTR) and the mean-time-

between-failures (MTBF). The diagnostic module can reduce the MTTR by helping the engineer pinpoint faults in the equipment. In addition, the module can predict impending malfunctions, thereby warning the operator to perform preventive maintenance before a catastrophic fault occurs. This can potentially extend the MTBF.

By predicting the final wafer state, the quality of the wafers at each process step can be classified to ensure that only wafers worth subsequent processing continue down the fabrication line. This also reduces the need for monitor wafers. Thus, the system has the potential to positively impact the yield, cycle time, and throughput of the fab as a whole [1.9].

Table 1.1 Cost of Ownership Impact by Each Module of System

Module	Ownership Cost Impact
Fault Detection	Reduce process scrap yield
Fault Diagnosis	Reduce the equipment down time
Wafer State Prediction	Determine value of wafer at process step

1.4 Thesis Organization

Chapter 2 discusses the real-time data and collection systems used in this work. A description of the experiments conducted to both develop and verify the algorithms presented in this thesis follows in Chapter 3. Chapters 4 through 6 develop the theory and show applications for each of the three modules which make up the overall Equipment Analysis and Wafer State Prediction System. Specifically, these are the Fault Detection, Fault Diagnosis, and Wafer State Prediction modules. Finally, conclusions and future directions for this research are given in Chapter 7.

References for Chapter 1

[1.1] T. E. Seidel, "Silicon Technology Roadmap to 2010," *International Symposium on Semiconductor Manufacturing (ISSM)*, Tokyo, June 1994, pp. 14-17.

[1.2] P. K. Chatterjee, G. B. Larrabee, "Gigabit Age Microelectronics and Their Manufacture," *IEEE Trans. Very Large Scale Integration (VLSI) Systems*, vol. 1, no. 1, Mar. 1993, pp. 7-21.

[1.3] W. T. Siegle, "Required Manufacturing Capabilities for Competitive Device Manufacturers in the Year 2000," *International Symposium on Semiconductor Manufacturing*, Tokyo, June 1994, pp. 6-9.

[1.4] M. Clayton, *private communication*, Aug. 1993.

[1.5] M. A. Lieberman, R. A. Gottscho, "Design of High Density Plasma Sources for Materials Processing," UCB/ERL M93/3, Jan. 11, 1993.

[1.6] J. P. McVittie, J. C. Rey, A. J. Bariya, M. M. IslamRaja, L. Y. Cheng, S. Ravi, K. C. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *SPIE: Advanced Techniques for Integrated Circuit Processing*, vol. 1392, 1990. pp. 126-138.

[1.7] V. Vahedi, R. A. Stewart, M. A. Lieberman, "Analytic Model of the Ion Angular Distribution in a Collisional Sheath," *J. Vac. Sci. Technol. A.*, vol. 11, no. 4, Jul/Aug 1993, pp. 1275-1257.

[1.8] "SEMATECH Applications of Cost-of-Ownership Models," SEMATECH, 92111368A-TR, May 18, 1993.

[1.9] S. Cunningham, C. J. Spanos, K. Voros, "Semiconductor Yield Improvement: Results & Best Practice," UC Berkeley Engineering Systems Research Center, ESRC94-14/CSM-10, Sept. 1994.

Chapter 2

Real-Time Tool Data

2.1 Introduction

The success of the Equipment Analysis and Wafer State Prediction System relies heavily on the data used for analysis. Therefore, the data used in the system must be carefully considered. The most direct solution is to actually measure the etch rate, selectivity, and anisotropy while the wafer is being processed. This capability, however, is not yet available. Therefore, empirically based models are used to predict the wafer outcomes. The choice of signals which best reflect the equipment performance is not obvious, especially for complex equipment such as the plasma etcher.

Much of the past work involving the modeling of plasma etch equipment has used classical response surface methodology (RSM) models which map the input settings, such as the radio frequency (RF) power, chamber pressure, gas flows, and electrode gap spacing, directly to the output states including the etch rate, uniformity, selectivity, and anisot-

ropy [2.1][2.2]. A limitation with this approach is that for plasma etchers, the same input settings do not always result in the same output wafer characteristics because the input settings of the machine are not closely coupled with the actual chamber state. In addition, drift in the machine from natural aging is not accounted for in these models [2.3]. Equally important, errors such as miscalibrated components will not be detected by examining the input settings of the machine. For example, if a mass flow controller (mfc) is miscalibrated, the controllers inside the equipment will not detect the error since the gas flow reading will appear to be correct even though the actual flow may not be within the desired specifications. This change in gas flow may subsequently lead to changes in the plasma characteristics, which in turn impact the etching process.

Because the machine input settings usually do not exert enough direct control over the desired outcome, there has been a push to use other sensors besides the input settings to monitor the equipment. Spanos *et al.* showed that the electrical and mechanical signals associated with the plasma RF can be modeled with time series models and used effectively to detect malfunctions [2.4]. Anderson used optical emission spectroscopy and partial least square regression (PLSR) techniques to model plasma wafer characteristics such as etch rates, selectivities, and uniformity [2.5]. Butler and Stefani performed run-to-run control of polysilicon gate etch using *in situ* spectral ellipsometry [2.6]. A group at M.I.T. is developing a full wafer monitoring system using interferometric imaging to determine the etch rate, selectivity, and uniformity across an entire wafer [2.7]. The underlying theme is that researchers are investigating signals which are more accurate than the input settings in describing the wafer states of interest.

The Equipment Analysis and Wafer State Prediction System presented in this thesis uses non-invasive real-time equipment signals. This thesis shows that for plasma processes in particular, electrical and mechanical signals such as the load impedance and coil

positions give a more accurate depiction of the chamber state than the input settings and can be used effectively for fault analysis and prediction of the final wafer state [2.3][2.8][2.9].

This chapter first details the process by which the real-time signals used for analysis are chosen. Next, the specific collection systems used for the plasma etch examples are described, followed by a discussion of the real-time data chosen for each type of etcher. In this work, two different types of state-of-the-art plasma etchers are investigated, a parallel plate system and an transformer coupled plasma (TCP) system. These etchers were selected because they are currently among the most advanced plasma etchers used in the semiconductor manufacturing industry. Due to different hardware considerations, the real-time data collected from the systems differ.

2.2 Real-Time Signal Selection

Because the Equipment Analysis and Wafer State Prediction System depends upon the data used for analysis, selecting the real-time signals most sensitive to the equipment state is critical. A typical etcher, for example, has well over 400 signals from which to choose [2.10]. These include signals involved in all steps of the process, from wafer handling to the pumping of the chamber to the power delivered by the RF generator. Many signals obviously do not directly affect the chamber, such as the signal determining whether or not the input cassette is lowered. The case is not as clear, however, for many other signals.

We do not monitor those signals which are tied directly to the input settings of the machine. Instead, signals which give the most information about the chamber state are monitored. For example, in plasma systems we do not include the RF power delivered from the generator for several reasons. First, any deviation in the RF power will be caught by the machine's own feedback loop. Second, if the RF generator is miscalibrated, this

signal will appear to have the proper values even if the incorrect power is delivered. This is similar in idea to the mass flow controller example in the previous section. Third, the value of the RF power read by the machine is before the matching network, which is not as accurate as the actual power delivered to the upper electrode of the etcher. For a typical etching process, we have found that the difference between the RF power before and after the matching network can be up to 10%. Therefore, instead of monitoring the RF power delivered by the generator, we measure the RF power at the upper electrode.

To test the relevance of the remaining chosen signals to the wafer state, the following steps are taken. First, a standard factorial experiment in which the input settings are varied over the range of the operating space of the equipment is conducted. The purpose of the experiment is to change the state of the equipment in a way that will affect the processing of the wafer. In addition to the factorial experiment, we run several wafers through the machine at the normal input settings, or baseline conditions, to obtain the "normal" fluctuations of the real-time signal readings. Then the ranges of the real-time signals collected during the factorial experiment and those collected during the baseline runs are compared. Those signals which have a substantial range relative to the baseline data are considered to be "sensitive" to the equipment state. More formally, an F-test is calculated:

$$\frac{s^2_{fact}/\nu_{fact}}{s^2_{cent}/\nu_{cent}} \sim F_{\alpha,\nu fact,\nu cent} \tag{2.1}$$

where s^2_{fact} is the estimated variance of the signal collected during the factorial experiment, s^2_{cent} is the estimated variance of the signal collected during the centerpoint runs, ν_{fact} is the degrees of freedom in the factorial experiment, and ν_{cent} is the degrees of freedom in the centerpoint runs. Those signals which have F-statistics above a desired level of significance are collected by the monitoring systems and used in the Equipment Analysis and Wafer State Prediction System.

2.3 Real-Time Collection Systems

The data collection systems used in the Equipment Analysis and Wafer State Prediction System monitor both electrical and mechanical signals in real-time. The two collection systems are: (1) the Brookside LamStation software, which reads the signals from the SECS-II (SEMI Equipment Communication Standard-II) serial port on the etcher [2.11] and (2) the Comdel Real Power Monitor (RPM-1), which reads the signals through its own RS232 interface [2.12]. Figure 2.1 depicts the Comdel monitor in relation to the RF generator, the matching network, and the upper and lower electrodes.

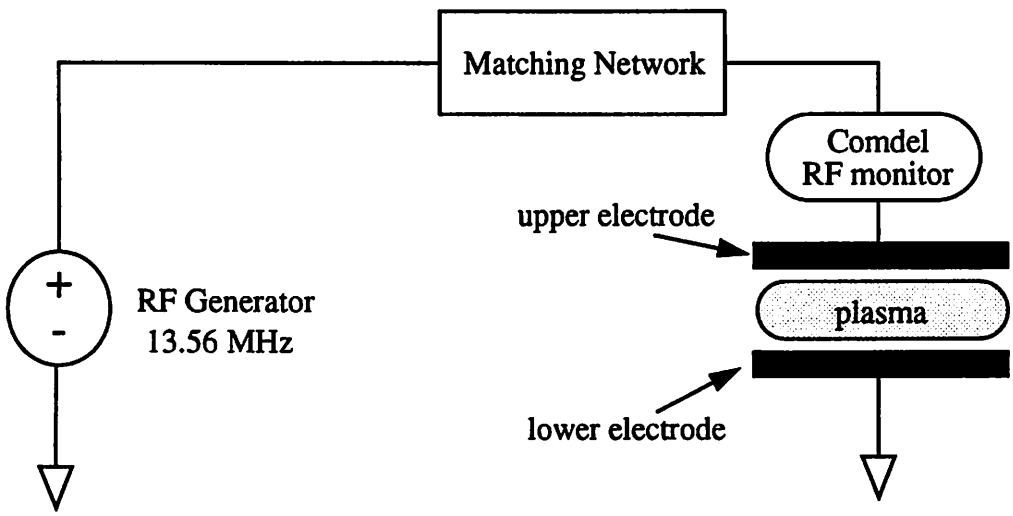


Figure 2.1 Position of the Comdel monitoring system relative to the matching network, power supply, and electrodes for a parallel plate reactor.

Plasma etch equipment have well over 400 signals from which to choose [2.10]. These range from the entrance load lock vacuum sensors, to electrical characteristics of the chamber, to the state of the gas flow valves. For fault detection, diagnosis, and wafer state prediction purposes, only a small subset is required. As detailed in the next section, the signals corresponding to the RF network are most sensitive to changes in the equipment

state [2.4]. Sample frequencies of 1 Hz have been achieved and were found to be sufficient for this application.

Since the Comdel RPM-1 resides after the matching network directly above the upper electrode, it gives more accurate readings of the forward power and other electrical parameters delivered to the plasma than does the LamStation software. The Comdel RPM-1 monitors the current, voltage, and DC bias at the upper electrode. Calculated from these values are the delivered RF power, root-mean-square (RMS) current and voltage values, RF impedance, and the phase angle between the current and voltage.

Because most of the signals collected are directly tied into the electrical or mechanical components of the machines, different sets of data are collected depending on the hardware configurations of the etcher. In this work, both the single wafer parallel plate Lam Rainbow 4400 polysilicon and inductively coupled Lam TCP 9600 metal plasma etchers are studied. While both the LamStation software and the Comdel RPM-1 RF probe are used to collect data from the parallel plate etching systems, only the LamStation software is used to collect data from the metal TCP system.

2.3.1 Parallel Plate Etching System

For parallel plate etching systems, between six and thirteen of the collected signals are used. Six signals are collected or calculated via the Comdel RPM-1. The other seven signals are collected via LamStation. The signals collected for the Lam Rainbow 4400 polysilicon plasma by each monitoring system are listed in Table 2.1. The important signals monitored are: RF Power, RF Voltage, RF Current, Load Impedance, RF Phase Error, DC Bias, RF Tune Vane Position, RF Load Coil Position, Peak-to-Peak Voltage, and Endpoint Data. Each of the above signals is described in Table 2.2. These signals were chosen because they are sensitive to changes in the state of the chamber of the etcher, which directly impacts the wafer. Because these measurements are related electrically or

mechanically, some signals are highly correlated. Three signals, Load Impedance, Phase Error, and DC Bias, are collected from different places in the equipment by the two independent monitoring systems. Although these readings are correlated, they are not identical.

Table 2.1 Real-Time State Signals Collected for the Lam Rainbow 4400

LamStation Software	Comdel RPM-1
RF Load Coil Position	RF Power
RF Tune Vane Position	RF Voltage
Peak-to-Peak Voltage	RF Current
Load Impedance	Load Impedance
RF Phase Error	RF Phase Error
DC Bias	DC Bias
Endpoint	

Table 2.2 Description of the Real-Time Signals

Signal	Description
RF Tune Vane Position	Position of the tune vane in the matching network of the upper electrode; acts as a variable capacitor
RF Load Coil Position	Position of the load coil position in the matching network of the upper electrode; acts as a variable inductor
RF Load Impedance	Apparent input impedance of the matching network
RF Phase Error	The phase error between the current and voltage (ideally 90^0) at the upper electrode
DC Bias	Measures the potential difference of the electrodes
Peak-to-Peak Voltage	Magnitude of voltage on the electrodes
End Point Data	Reads the intensity of the plasma in the chamber at a particular wavelength
RF Voltage	Root-mean-square (RMS) voltage at the upper electrode
RF Current	RMS current at the upper electrode

2.3.2 TCP Etching System

The signals of interest for the Lam TCP 9600 metal etcher are slightly different from those of the parallel plate system since the plasma source of the two systems differ. Instead of upper and lower electrodes, the TCP source consists of planar coils wound from the center to the outer radius of the source chamber, one placed at the top of the chamber, the other at the bottom [2.13]. The plasma is created when the gas near the coil ionizes as a result of the induced RF electric field. Similar to the parallel plate system, TCP sources can be driven at 13.56 MHz. Since these systems run at lower pressures and generally produce higher density plasma than parallel plate systems, they are claimed to produce more anisotropic etches with smaller linewidths and faster etch rates.

Because the RPM-1 is not suited for the TCP source, the analysis for the TCP machine was based solely on the data collected from the LamStation software. Many signals similar to those used for the parallel plate systems are collected for the TCP system, with a few additional signals. The signals which best reflect the equipment state are related to both the bottom and upper coils. The signals associated with the bottom coil are similar to those for the parallel plate system: RF Tune Vane Position, RF Load Coil Position, Line Impedance, RF Phase Error, and DC Bias. As in the parallel plate system, these signals are used to tune the matching network. Similar signals are collected from the matching network of the top coil. For example, instead of a tune vane, a tune capacitor is used. One of the most sensitive signals to process changes is the RF Bias, which measures the DC bias between the top and bottom sources when both are powered. As in the parallel plate systems, endpoint information is also collected. A description of the real-time signals collected for the Lam TCP 9600 metal etcher is given in Table 2.3.

Table 2.3 Real-Time State Signals Collected for the Lam TCP 9600

	LamStation	Description
Bottom TCP Coil	RF Tune Vane Position	Equivalent position of the tune vane position in matching network of the lower coil
	RF Load Coil Position	Equivalent position of the load coil position in matching network of the lower coil
	Line Impedance	Apparent input impedance of the lower matching network
	RF Phase Error	Phase error between the current and voltage at the bottom coil
	DC Bias	Measures the charge on the electrodes
Top TCP Coil	TCP Tune Vane Capacitor Position	Position of the tune vane capacitor of the matching network for the top coil
	TCP Phase Error	Phase error between the current and voltage at the top coil
	TCP Load Capacitor Position	Position of the load capacitor of the matching network for the top coil
	Line Impedance	Apparent input impedance of the upper matching network
	RF Bias	DC bias when both sources are powered
	Endpoint	Reads the intensity of the plasma in the chamber at a particular wavelength

2.4 Pre-filtering of Real-time Data

The raw data collected for both types of etchers includes several peripheral steps in the etching procedure, such as the stabilization of the pre-etch gases, the pre-etch in which the native oxide is etched away, the stabilization of the main etch gases, the main etch, and finally the unloading of the wafer from the chamber. The step of interest in this application is the main etch step. Although value is gained from examining the loading and stabilization steps for gas leaks for example, this work focuses on fault detection and diagnosis

during the main etch step and determines how the wafer state is impacted. The algorithms presented can be extended to include the other windows of operation.

The signals collected during the main etch step are concatenated and then filtered as follows. Characteristics of the real-time signals caused by transient effects during processing must be accounted for before statistical analysis. At the beginning of the main etch step for each wafer, for example when RF power is applied, a small transient occurs while power is stabilizing. The analysis is delayed a few seconds until the signals have stabilized. The delay time, based on the stabilization time for a normally processed wafer, is illustrated in Figure 2.2. If the signal does not stabilize within the specified time, the Fault Detection Module generates an alarm, as discussed in Chapter 4. To simplify the calculations in each module, the same number of data points for each collection system is used for each wafer.

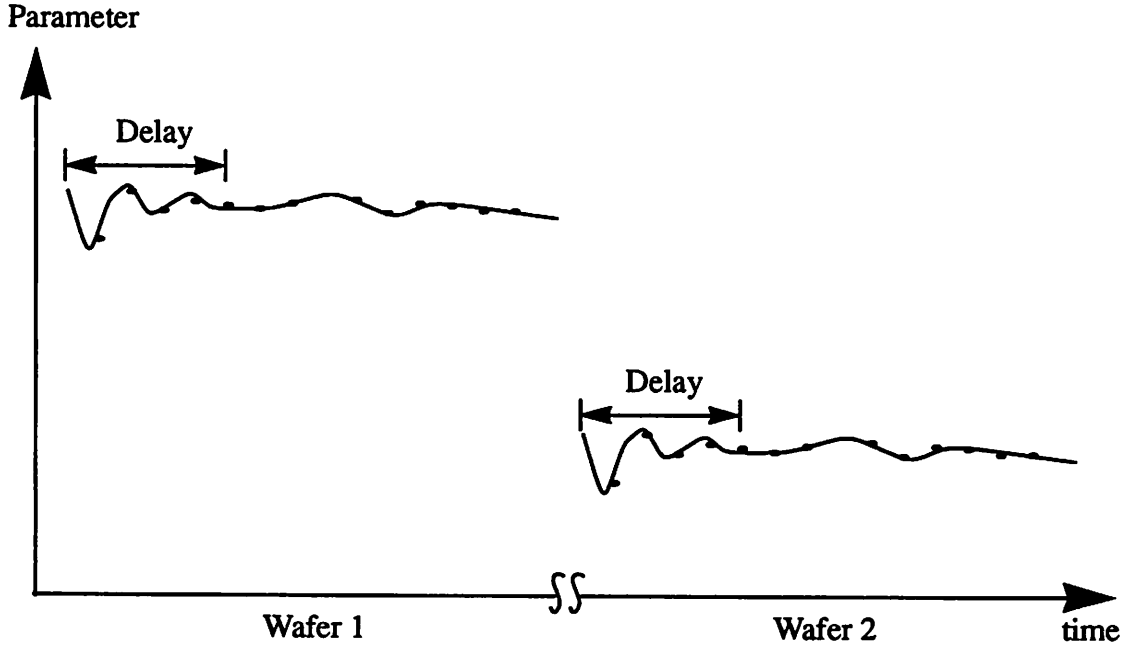


Figure 2.2 Real-time Signal Filtering

2.5 Real-Time Data Example

Approximately 30 points are collected per signal per wafer etch for the LamStation data and 50 points for the RPM-1 data. Since the data are collected sequentially at a sampling rate of 1 Hz for the LamStation data and at 2 Hz for the RPM-1 data, the real-time signals are autocorrelated in time and demonstrate time series behavior. Time series patterns are observed both within each wafer and across several wafers due to controller adjustments and equipment aging. The time series nature of the data is exploited for fault detection, as will be shown in Chapter 4.

Figure 2.3 shows the real-time signals of the RF Load Coil Position and DC Bias for different fixed input conditions on each of 12 wafers, collected by the LamStation software. Notice the instability in wafers #4 and #5 shown in Figure 2.3. (These wafers are identified as "faulty" by the Fault Detection Module described in Chapter 4.) For an unknown reason, the RF power dropped significantly during the processing of wafer #4, causing corresponding adjustments in both Coil Position and DC Bias. Later measurements show that the etch rate for wafer #4 was unusually low due to the drop in RF power. Therefore, the run corresponding to wafer #4 was rejected from the analysis. As seen in Figure 2.3, wafer #5 exhibited unstable signals and was also rejected as an outlier. Excluding wafers #4 and #5, Figure 2.3 also shows that the wafer-to-wafer variance is much larger than the within-wafer variance.

Figure 2.4, which shows the Load Impedance and RF Tune Vane Position for the duration of six wafers processed at the same input settings, illustrates that the real-time signals chosen reflect equipment state better than the input settings. While the input settings are fixed for all six wafers, the real-time signals vary for each etch, indicating that the specific real-time data described in the previous sections give a more accurate description of the

actual equipment state. As a consequence, the real-time data can be used effectively for fault detection, diagnosis, and prediction of wafer states.

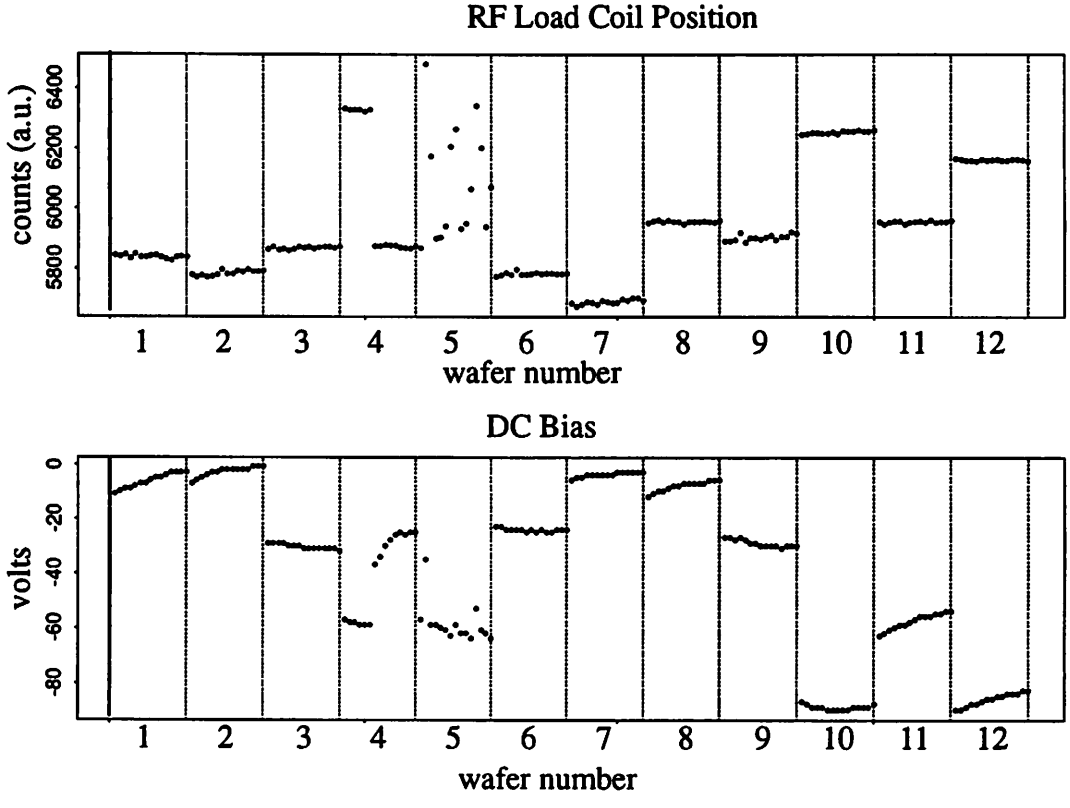


Figure 2.3 Real-time signals of RF Load Coil Position and DC Bias for different input conditions on 12 wafers. Wafers #4 and #5 have unstable real-time signals and are rejected as "bad" wafers [2.8]. Notice the large wafer-to-wafer variance compared to the within-wafer variance.

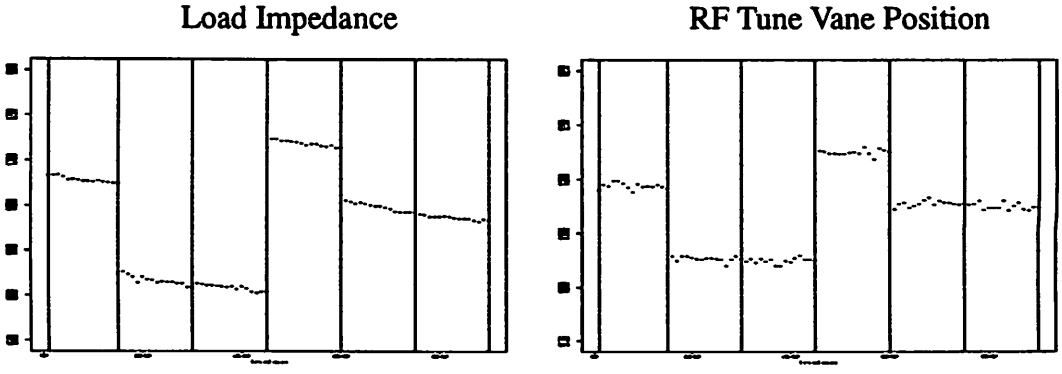


Figure 2.4 Real-time signals for six wafers processed with idential input settings during the duration of the main etch. Unlike the fixed input settings, the real-time signals reflect changes in machine state.

References for Chapter 2

[2.1] G. S. May, J. Huang, C. J. Spanos, "Statistical Experimental Design in Plasma Etch Modeling," *IEEE Trans. Semiconductor Manufacturing*, vol. 4, no. 2, May 1991, pp. 83-98.

[2.2] K. J. McLaughlin, S. W. Butler, T. F. Edgar, I. Trachtenberg, "Development of Techniques for Real-Time Monitoring and Control in Plasma Etching: I. Response Surface Modeling of CF_4/O2 and CF_4/H2 Etching of Silicon and Silicon Dioxide," *J. Electrochem. Soc.*, vol. 138, no. 3, March 1991, pp. 789-799.

[2.3] S. F. Lee, C. J. Spanos, "Prediction of Wafer State After Plasma Processing Using Real-Time Tool Data," submitted to *IEEE Trans. Semiconductor Manufacturing*.

[2.4] C. J. Spanos, H. Guo, A. Miller, J. Levine-Parrill, "Real-Time Statistical Process Control Using Tool Data," *IEEE Trans. Semiconductor Manufacturing*, November 1992. pp. 308-318.

[2.5] H. M. Anderson, M. P. Splichal, "An Integrated System of Optical Sensors for Plasma Modeling and Plasma Process Control," *Proc. of SPIE*, vol. 2091, Monterey, CA, Sept. 27-29, 1993, pp. 333-334.

[2.6] S. W. Butler, J. A. Stefani, "Supervisory Run-to-Run Control of Polysilicon Gate Etch Using *In Situ* Ellipsometry," *IEEE Trans. Semiconductor Manufacturing*, vol. 7, no. 2, May 1994, pp. 184-192.

[2.7] D. S. Boning, J. L. Claman, K. S. Wong, T. J. Dalton, H. H. Sawin, "Plasma Etch Endpoint via Interferometric Imaging," *Proceedings of the American Control Conference*, June 1994, pp. 897-901.

[2.8] S. F. Lee, E. D. Boskin, H. C. Liu, E. Wen, C. J. Spanos, "RTSPC: A Software Utility for Real-Time SPC and Tool Data Analysis," to appear in *IEEE Trans. Semiconductor Manufacturing*.

[2.9] C. J. Spanos, S. Leang, S. F. Lee, "A Control and Diagnosis Scheme for Semiconductor Manufacturing," *Proceedings of the American Control Conference*, June 1993, vol. 3, pp. 3008-3012.

[2.10] Lam Research, "Rainbow Plasma Etch Systems," *Operations and Maintenance Manuals*, vol. 1, revision 3.0, March 1992.

[2.11] *LamStation Rainbow*, version 3.6, Brookside Software, 1991.

[2.12] Real Power Monitor (RPM-1), Comdel Inc.

[2.13] M. A. Lieberman, R. A. Gottscho, "Design of High Density Plasma Sources for Materials Processing," UCB/ERL M93/3, Jan. 11, 1993.

Chapter 3

Experimental Design

3.1 Introduction

The experiments conducted to develop and verify the real-time Equipment Analysis and Wafer State Prediction System are discussed in detail in this chapter. As previously stated, the plasma etchers used in this work are a Lam Rainbow 4400 polysilicon etcher and a Lam 9600 metal etcher. First, the set of experiments performed in the Berkeley Microfabrication Laboratory on a parallel plate polysilicon etcher is described. These experiments, which form the basis for the majority of the system analysis in the following chapters, were designed to span a significant amount of time to allow for machine aging. Next, the experiments conducted on a TCP metal etcher, used to adapt the fault detection algorithm for multiple recipes, is discussed. Each experiment has its own test structure and measurement set, which will also be described.

3.2 Lam Rainbow 4400

This section focuses on the experiments conducted in the Berkeley Microfabrication Laboratory on a Lam Rainbow 4400. The purpose of this experiment was three-fold. The first goal was to verify the new fault detection algorithm to detect single faults at different levels of severity; the second was to develop and test the diagnostic capabilities for single faults; the third was to build models for several pertinent wafer states using the real-time data, and then test the predictive capability of the models with an independent data set. To achieve these goals three separate experiments, the Training, Verification, and Diagnostic Experiments, were conducted.

The Training Experiment is a central composite experiment composed of two phases [3.1]. Both phases are used to build the wafer state models. The second phase is used to both test the Fault Detection Module and train the Diagnostic Module. The Verification Experiment, run one month after the Training Experiment, is extremely important, as it provides an independent data set used to verify the Wafer State Prediction Module. In a month's time, the machine suffers from general wear and tear, such as chamber coating and electrode conditioning, which affect the performance of the equipment. In a high-volume manufacturing site where 5,000 wafers are processed every week, the condition of the chamber gradually changes with time. Therefore, it is important to verify that the prediction models survive these normal machine drifts. The Verification Experiment is also used to verify the algorithms used in the Diagnosis Module. Because two different data sets are used, one for training and the other for testing, the actual predictive and diagnostic capabilities of the modules are determined. The Diagnostic Experiment further tests the diagnosis algorithms. Since a different test structure was used for this experiment, it tests the application of the diagnostic algorithms for different wafer loadings.

3.2.1 Test Structures

Two different test structures were used, one for the Training and Verification Experiments, and the other for the Diagnosis Experiment. The layout of each test structure is discussed in this section, followed by a brief outline of the process flow.

3.2.1.1 Layout

The wafers for the Training and Verification Experiments are 4" diameter wafers patterned with polysilicon, gate oxide, photoresist, and low temperature oxide. Any exposed materials on the wafer change the chemical composition of the ionized gas. It has been observed that the etch characteristics depend not only on the type of layers etched, but also on the specific patterns created. This effect is known as loading. The test structure was designed so that the polysilicon, gate oxide, photoresist, and the LTO hard mask will be simultaneously etched in the same etch step. Due to complex loading effects, this results in more accurate etch rates and selectivities than etching blanket wafers individually. The wafer states of interest are the etch rate of polysilicon, selectivity of polysilicon to gate oxide (ratio of the polysilicon etch rate to the gate oxide etch rate), selectivity of polysilicon to I-line positive photoresist (ratio of the polysilicon etch rate to the positive photoresist etch rate), and the non-uniformity of the polysilicon etch. Due to the small ranges of selectivities across the design space, models are created for the individual etch rates of gate oxide and photoresist. The test structure, requiring a three mask process, allows all models to be developed from the same set of experimental conditions [3.6]. Figure 3.1 shows a simplified view of the test structure indicating all of the surfaces that were etched during the Training and Verification Experiments.

Due to technical difficulties with the oxide etcher in the Microfabrication Laboratory, the process for the Diagnosis Experiment was simplified to a single mask process. As a

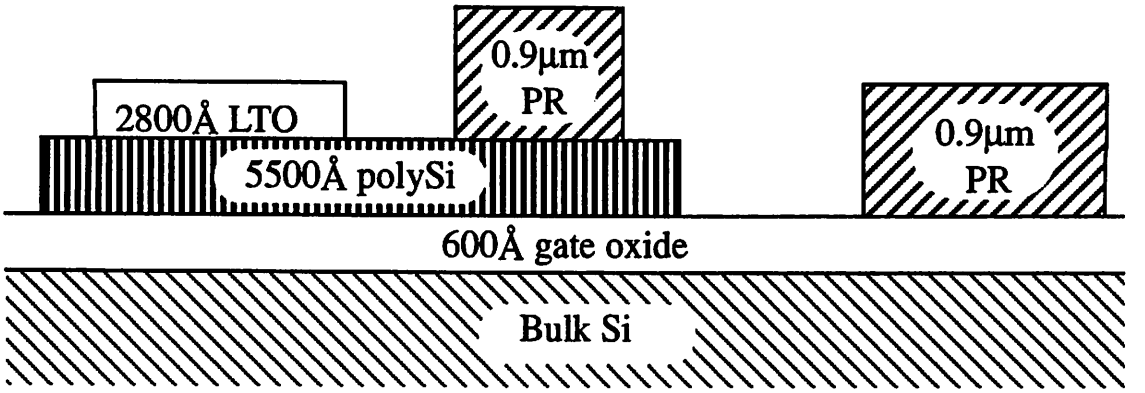


Figure 3.1 Test structure for the Training and Verification Experiments.

result, only polysilicon and photoresist were exposed to the etch during this experiment, as shown in Figure 3.2.

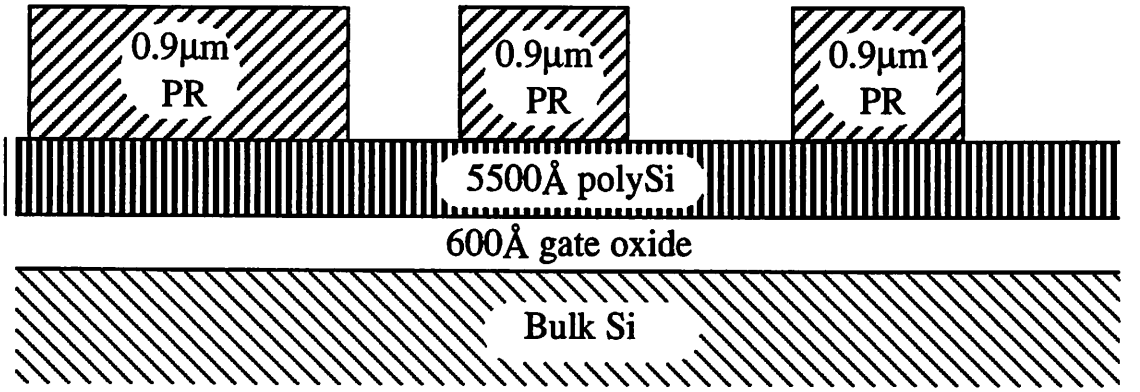


Figure 3.2 Test structure for the Diagnosis Experiment.

3.2.1.2 Process Flow

A 600Å thermal oxide was first grown on the wafers ("oxide" refers to silicon dioxide, SiO_2) followed by 6000Å n+ doped polysilicon, deposited via low pressure chemical

vapor deposition (LPCVD). After a 20 minute nitrogen anneal at 950°C, 2800Å undoped low temperature oxide (LTO) was deposited by chemical vapor deposition.

Three mask steps are required to build the test structure for the Training and Verification Experiments. The first mask defines the gate oxide open regions, and requires both oxide etching through LTO and polysilicon etches that terminates when the gate oxide is exposed. Approximately 50Å of gate oxide was etched during the polysilicon etch and overetch. The second mask defines the polysilicon open areas, requiring an LTO etch that terminates when the polysilicon is exposed. Since the selectivity of LTO to polysilicon is low in the Lam Autoetch SiO_2 Etcher, about 500Å polysilicon was etched in this step. The last mask defines the areas with photoresist. (Photoresist is generally used to define the patterns on the wafer. The regions not covered with photoresist are etched away. The covered regions are protected from being etched, and are therefore retained.) A more detailed description of the process is listed in Appendix A. The process flow for the Diagnosis Experiment test structure, a subset of the above process, follows steps # 1 - 5 outlined in Appendix A.

3.2.2 Training Experiment

The standard polysilicon plasma etch contains a pre-etch step which etches through the native oxide layer. (About 20Å of native oxide is grown naturally on wafers when exposed to air. It is a "crud" oxide layer that must be stripped off the wafer.) The pre-etch is followed by the main etch step, which is the step of interest in this project. Most of the significant etching occurs during this main etch step. In this experiment, the pre-etch recipe was constant for all etches, while the main etch recipe was modified. To obtain more accurate etch rates and thus better selectivity measurements, the main etch, called the centerpoint etch, was a timed etch. The pre-etch and centerpoint recipes are listed below in Table 3.1.

Table 3.1 Etch Recipes

Input Parameter	Pre-etch	Centerpoint
Pressure (mtorr)	400	425
Power (Watts)	200	275
Gap (cm)	1.0	0.9
Cl_2 (sccm)[a]	0	160
SF_6 (sccm)	100	0
He (sccm)	0	380
He clamp (torr)	8.0	8.0

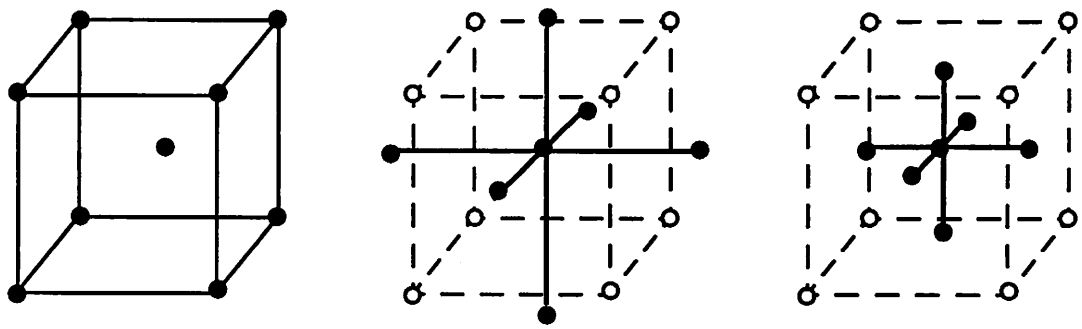
a. The flow rates are in units of sccm, "standard cubic centimeters per minute"

Given the above recipes, the input parameters varied in the experiment are: Pressure (P), Power (W), Gap (G), Gas ratio of Cl_2 to He (R), and the Total gas flow of Cl_2 and He (T). Note that because the gas ratio and total gas flows are more significant to the etch results, they were varied in the experiment instead of the individual gas flows. As previously stated, the output wafer states, or responses to the experiment, were the etch rate of polysilicon, selectivity of polysilicon to oxide and I-line positive photoresist, and polysilicon wafer non-uniformity.

The Training Experiment consisted of two phases. Phase I is the variable screening stage, which determines which variables are statistically significant in the models. Phase II assesses the quadratic nature of the system via a star design [3.1]. The input values used for all experiments are listed in Table 3.2, in terms of percent offset from the nominal values. Figure 3.3 illustrates the different points for three parameters in the input space covered by the Training and the Verification Experiments. The particular values were chosen to cover a wide range of operating conditions of the machine. The next two subsections describe the two phases of the Training Experiment.

Table 3.2 Change in Percent From Nominal

Parameter	Training Experiment		Verification Experiment
	Phase I	Phase II	
Pressure	±15%	±22.5%	±10%
Power	±15%	±22.5%	±10%
Gap	±11%	±17%	±10%
Flow Ratio	±19%	±22%	±10%
Total Flow	±11%	±22%	±10%



(a) (b) (c)

Figure 3.3 Depiction of three input settings for the (a) Training Phase I, (b) Training Phase II, and (c) Verification experiments.

3.2.2.1 Phase I: Variable Screening

Phase I consists of a two-level, 16 run fractional factorial design and 4 center points. This is a design of resolution V with no blocking, but drops to resolution III when blocking for time and for the fact that the wafers came from two different lots. Since blocking was not a factor in any of the phase I response surface models, the design is essentially of resolution V. Thus, no main effects are confounded with one another or with second or third order effects. Additionally, second order effects are not confounded with one another. Main effects are, however, confounded with fourth order effects, and second

order effects are confounded with third order effects. Assuming that the fourth order effects are negligible, this experiment provides a good estimate of the main effects.

As mentioned above, there are two blocks, one for time and one for using two different lots. The time block is confounded with the three-way interaction among the gas ratio, power, and electrode gap spacing, or equivalently, the two-way interaction between the chamber pressure and total flow. The effect of using two different lots is confounded with the three-way interaction among the chamber pressure, power, and electrode gap spacing, which is equivalent to the two-way interaction between the gas ratio and total flow. Although all 20 runs of the experiment were run in one day, the block for time was a precaution in the event the etcher malfunctioned in the middle of the experiment.

The lots were also blocked due to the way the wafers were processed before the etch step. In particular, the lot non-uniformity[1] of the polysilicon across a wafer lot grown via LPCVD was poor. The lot non-uniformity across two boats[2], or 24 total wafers, was 41%, compared with approximately 9% within one boat of 12. The average within wafer non-uniformity[3] in the one boat was 2%. In addition to the degradation of the lot uniformity, the sheet resistance across two boats suffered as well, showing non-uniformity of 45%. Across one boat the sheet resistance non-uniformity was reduced by half, to 19%. Therefore, to reduce the variation across the experiment wafers, only one boat of 12 wafers was deposited with polysilicon at a time. While both sets of wafers were annealed at the same time, LTO also was deposited one boat at a time for better uniformity within the boat. Therefore, the experimental wafers in phase I were blocked for the two different deposition lots.

---

1. Lot non-uniformity is calculated as the difference of the average deposition rate between the first and last wafers, divided by the average deposition rate across all wafers.

2. Wafers in deposition furnaces are placed in "boats," which hold the wafers vertically a fixed distance apart from one another.

3. Within wafer non-uniformity is calculated as the difference of the average deposition rate near the edge of the wafer and the average deposition rate near the center of the wafers, scaled by the etch rate of the center average.

The object of the experiment was to examine the output space given a reasonable range of input settings for the main etch. The ratio of gases and total gas flow ranges were constrained by the limits on the mass flow controllers of the He and Cl_2which are 500 sccm and 200 sccm, respectively. Given these limits, the design ranges chosen for the gas ratios and total flows are as large as possible. Table 3.3 shows the values of the input parameters (+, -) used in the two-level fractional factorial design. The middle level (0) shows the values for the center-point recipe. The actual flow values corresponding to the ratio of gases and total flow in the experiment are listed below in Table 3.3.

Table 3.3 Values for Cl_2 and He

	Gas	Total Flow = 600 sccm	Total Flow = 480 sccm
Ratio = 0.50	Cl_2	200	160
	He	400	320
Ratio = 0.34	Cl_2	152	122
	He	448	358

In the first phase, the values for pressure and power are 15% offset from the nominal values, gap and total flow values are 11% from the nominal values, and the ratio values are 19% from the nominal value. Before performing the experiment the runs are randomized in the time blocks including four centerpoint runs to check for non-linearity. The actual runs conducted, in order of execution, are listed in Table 3.4 and Table 3.5.

Table 3.4 Randomized Phase I Block I Runs

run #	P	R	W	G	T	lot	wfr#
12	489	0.50	234	1.0	480	8-1	1
2	489	0.34	234	0.8	480	8-2	23
C1	425	0.42	275	0.9	540	8-1	5

Table 3.4 Randomized Phase I Block I Runs

run #	P	R	W	G	T	lot	wfr#
1	361	0.34	234	0.8	600	8-1	6
14	489	0.34	316	1.0	480	8-2	13
7	361	0.50	316	0.8	600	8-2	17
8	489	0.50	316	0.8	480	8-1	12
11	361	0.50	234	1.0	600	8-2	21
C2	425	0.42	275	0.9	540	8-2	19
13	361	0.34	316	1.0	600	8-1	3

Table 3.5 Randomized Phase I Block II Runs

run #	P	R	W	G	T	lot	wfr#
9	361	0.34	234	1.0	480	8-2	22
C3	425	0.42	275	0.9	540	8-2	16
15	361	0.50	316	1.0	480	8-1	2
5	361	0.34	316	0.8	480	8-2	20
10	489	0.34	234	1.0	600	8-1	10
3	361	0.50	234	0.8	480	8-1	7
C4	425	0.42	275	0.9	540	8-1	11
16	489	0.50	316	1.0	600	8-2	15
6	489	0.34	316	0.8	600	8-1	4
4	489	0.50	234	0.8	600	8-2	18

After examining the real-time signals, it was noticed that a few wafers from the first phase experienced equipment faults, such as instabilities in RF power, or phase error. The affected run numbers are # 1, 5, 10, 14, and 15. These runs were repeated, as shown in Table 3.6.

Table 3.6 Replicated Runs of Phase I

run #	P	R	W	G	T	lot	wfr#
1	361	0.34	234	0.8	600	9-1	7
5	361	0.34	316	0.8	480	9-2	20
10	489	0.34	234	1.0	600	8-2	24
14	489	0.34	316	1.0	480	9-1	19
15	361	0.50	316	1.0	480	9-2	10

The replicated runs #1 and #14 resulted in stable signals, but those for runs #5, 10, and #15 remained unstable. It is possible that the settings for those runs put the machine in an unstable state. Therefore, these points were not used in the subsequent models or analysis.

The screening analysis was performed by building models using the input settings. Statistical significance of each parameter was determined via the student-t test at the 0.05 significance level. Results of the analysis show that although all input settings are not statistically significant in each model, they are all required to model the output characteristics of interest. Listed in Table 3.7 are the t-values and p-values of models for the etch rates of polysilicon, oxide, and photoresist using the Phase I data. All the main effects and two-way interactions were used to build the models. Only the values for significant coefficients at the 0.05 level are listed. Surprisingly, pressure and ratio are not significant for the polysilicon etch rate model. The model for photoresist etch rate, on the other hand, requires all of the main input settings. The precise effect of each parameter on the output characteristics is better modeled after the Phase II experiment described below.

3.2.2.2 Phase II: Modeling Non-Linear Effects

In Phase II, additional runs were performed to determine the quadratic behavior of the system. The model is limited to quadratic terms because as the order of the polynomial

Table 3.7 Significance Tests for Phase I Models

Parameter	Polysilicon		Oxide		Photoresist	
	t value	p value	t value	p value	t value	p value
Pressure (P)					4.0095	0.0070
Ratio (R)			-5.3866	0.0004	9.3848	0.0001
Power (W)	16.7520	0.0000	19.3474	0.0000	45.2959	0.0000
Gap (G)	-9.9213	0.0000	-9.0391	0.0000	-13.4383	0.0000
Total (T)	2.6182	0.0307			4.8966	0.0027
P*R			3.6911	0.0050	-12.7136	0.0000
P*W	6.5525	0.0002	4.3588	0.0018		
P*G	-3.8305	0.0050			-14.0121	0.0000
P*T	-5.1635	0.0009	-4.7697	0.0010		
R*W					20.0163	0.0000
R*G	-6.4071	0.0002	-3.8825	0.0037	-10.3075	0.0000
R*T					-14.0674	0.0000
W*G	-2.7024	0.0270	-2.7276	0.0233	-6.3457	0.0007
W*T	2.3032	0.0502				
G*T						

increases, so does the number of terms required in the model. Thus, as long as the model fit is reasonable, only linear and quadratic terms will be used in the models [3.1].

The additional runs consist of center points and "star" points, arranged symmetrically along the axis of each variable (see Figure 3.3). For each variable two star points are run. Two center points were run, making a total of 12 additional runs. When arranged properly, these star points are orthogonal to each of the columns of Phase I. Therefore, the quadratic nature of the model can be estimated, even if a level shift occurred between Phases I and II [3.1].

Table 3.8 shows the star points for the Phase II runs. Table 3.9 and Table 3.10 show the corresponding gas flows for ratio and total flow. As listed in Table 3.2, each of the values for pressure and power are 22.5% offset from the nominal value, those for gap are 16.7%, ratio is 23%, and total flow is 22% from the nominal value. Once again, these values were chosen to achieve the widest operating range of the equipment.

Table 3.8 Star points

Parameter	Low value	High value
Pressure (mtorr)	329	521
Power (W)	213	339
Gap Spacing(cm)	0.75	1.5
Ratio	0.26	0.58
Total Flow (sccm)	420	660

The gas flows which correspond to the above table are as follows:

Table 3.9 Values for Cl_2 and He for Ratio Star Points

	Gas	Total Flow = 540 sccm
Ratio = 0.58	Cl_2	238
	He	342
Ratio = 0.26	Cl_2	112
	He	428

Table 3.10 Values for Cl_2 and He for Total Flow Star Points

	Gas	Ratio = 0.42
Total Flow = 660 sccm	Cl_2	195
	He	465

Table 3.10 Values for Cl_2 and He for Total Flow Star Points

	Gas	Ratio = 0.42
Total Flow = 420 sccm	Cl_2	124
	He	296

The randomized star and two center points run in Phase II are listed, in order of execution, in Table 3.11.

Table 3.11 Randomized Phase II Runs

run #	P	R	W	G	T	lot #	wfr #
25	425	0.42	339	0.9	540	9-1	4
28	425	0.42	275	0.75	540	9-2	16
24	425	0.26	275	0.9	540	9-2	14
C5	425	0.42	275	0.9	540	9-1	1
29	425	0.42	275	0.9	660	9-1	6
22	329	0.42	275	0.9	540	9-2	17
27	425	0.42	275	1.5	540	9-1	2
26	425	0.42	213	0.9	540	9-2	18
C6	425	0.42	275	0.9	540	9-2	13
23	425	0.58	275	0.9	540	9-1	5
30	425	0.42	275	0.9	420	9-2	15
21	521	0.42	275	0.9	540	9-1	3

3.2.3 Verification Experiment

The purpose of the Verification Experiment is to collect a second data set which can be used to test the prediction capability of the models. After the models are built with the data from the Training Experiment, they are tested with the data from the Verification Experiment to determine a prediction metric that indicates the overall prediction accuracy of the

models. The Verification Experiment was run about four weeks after the Training Phase II Experiment. The input settings for this experiment are ±10 % from one of the nominal values at a time. These runs were similar to the star points of Phase II, but at smaller deviations from the nominal values. Table 3.12 shows the run conditions for the Verification Experiment.

Table 3.12 Verification Experiment Runs

run #	P	R	W	G	T	lot #	wfr #
V1	383	0.42	275	0.9	540	10	1
V2	425	0.42	275	0.9	540	10	14
V3	425	0.42	247	0.9	540	10	13
V4	425	0.42	275	0.85	540	10	8
V5	425	0.42	275	0.9	540	10	6
V6	425	0.38	275	0.9	540	10	18
V7	425	0.42	275	0.9	513	10	20
V8	425	0.42	275	0.9	540	10	4
V9	467	0.42	275	0.9	540	10	7
V10	425	0.42	303	0.9	540	10	17
V11	425	0.42	275	0.9	540	10	2
V12	425	0.42	275	0.95	540	10	16
V13	425	0.42	275	0.9	540	10	19
V14	425	0.46	275	0.9	540	10	15
V15	425	0.42	275	0.9	567	10	5

3.2.4 Diagnosis Experiment

While the purpose of the previous experiments were to obtain data to develop and verify all three modules of the system, the objective of the Diagnosis Experiment was to obtain data sets to further verify the Diagnosis Module, described in detail in Chapter 5. In

this experiment, the RF power, chamber pressure, and Cl_2 gas flow were varied one at a time on the Lam Rainbow 4400 etcher. Recall that in the previous experiments, because the gas ratio and total flow of the gases are more suitable when modeling etch rates, they were varied instead of single gas flows. When the gas ratio was varied, the total flow was kept constant, and vice versa. When detecting and diagnosing equipment faults, however, it is more probable that one mass flow controller will be faulty at a time, so that the gas ratio or total flow will not remain constant when the other changes. Therefore, this experiments simulates a faulty mass flow controller by varying the Cl_2 gas flow alone.

The levels at which the single faults were injected are ±15 % and ±7.5 % from the same nominal values used in the previous experiments. Table 3.13 summarizes the input setting values for these percentages. Note that the He flow remained constant at 380 sccm and the gap spacing was fixed at 0.9 cm for all runs.

Table 3.13 Diagnosis Experiment: Input settings

Input Setting	- 15%	- 7.5%	Center-point	+ 7.5%	+ 15%
Pressure (mtorr)	361	393	425	457	489
Power (watts)	234	254	275	296	316
Cl_2 (sccm)	136	148	160	172	184

Runs at each setting were replicated, and four centerpoint wafers were run, making a total of 28 runs. Half of the runs were used to train the Diagnosis Module, and half were used to determine the accuracy of diagnosis. Blocks were chosen to account for equipment aging and chamber seasoning during the experiment, differences in the processing of the wafers, and different wafer lots. The runs for each block were then randomized. The resulting blocks are listed, in execution order, in Table 3.14 and Table 3.15.

Table 3.14 Diagnosis Experiment: Block I Randomized Runs

run#	P	W	Cl_2	lot#	wfr#
11	425	275	148	15	8
6	425	296	160	15	23
10	425	275	172	14	19
8	425	254	160	15	3
5	425	296	160	14	18
1	457	275	160	15	9
4	393	275	160	14	20
7	425	254	160	14	24
13	425	275	160	14	15
2	457	275	160	15	14
9	425	275	172	15	20
12	425	275	148	15	19
3	393	275	160	14	13

Table 3.15 Diagnosis Experiment Block II Randomized Runs

run#	P	W	Cl_2	lot#	wfr#
9	425	275	184	15	4
3	361	275	160	15	7
12	425	275	136	15	21
6	425	316	160	15	11
2	489	275	160	15	5
10	425	275	184	15	18
13	425	275	160	14	16
5	425	316	160	15	24
7	425	234	160	14	23
11	425	275	136	15	15

Table 3.15 Diagnosis Experiment Block II Randomized Runs

run#	P	W	Cl_2	lot#	wfr#
4	361	275	160	15	16
8	425	234	160	14	22
1	489	275	160	14	14

3.2.5 Wafer Measurements

In all the experiments for the Lam Rainbow 4400, film thickness measurements were taken by a Nanometrics Nanospec AFT system on 9 die per wafer. The points measured are as depicted in Figure 3.4. An index of refraction of 3.7 was used for polysilicon, 1.456 for oxide, and 1.631 for positive photoresist. The polysilicon measurements were taken over 600Å gate oxide, while the photoresist measurements were taken over 550Å gate oxide. The thinner gate oxide was due from initial endpoint etching of the polysilicon to clear area for the photoresist. The Alphastep 200 Automatic Step Profiler was used to double check the Nanospec measurements. The film thicknesses were measured before and after etching. Thicknesses of polysilicon, gate oxide, and photoresist were measure for those wafers etched in the Training and Verification Experiments, while only the thickness of polysilicon was measured for those etched in the Diagnosis Experiment.

The etch rates at each measured point were calculated by subtracting the post-etch from the pre-etch measurements, and dividing by the etch time. In the models, etch rates are averaged over the 5 points in the inner ring, as shown in Figure 3.4. The non-uniformity was calculated by taking the difference between the etch rates of the outer ring of 4 points and the inner ring of 5 points, scaled by the etch rate of the inner ring.

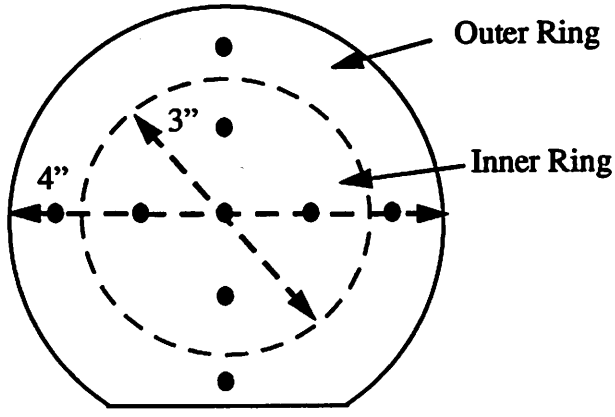


Figure 3.4 Wafer Measurement Points

3.3 Lam TCP 9600

The second set of experiments was performed on a Lam TCP 9600 metal etcher. We used the results of this experiment to develop and verify an algorithm to include various input recipes in the fault detection algorithm. The actual experiment was conducted by Texas Instruments in Dallas in the context of a larger study of various sensors and analysis techniques. The test structure used in this experiment was a multi-layer structure with TiN, Al, TiN, and oxide on silicon, which mimics the via and contact processes Texas Instruments is developing. A schematic is shown in Figure 3.5.

3.3.1 Static Experiment

The first experiment conducted by Texas Instruments was a three-level, fractional factorial design with six centerpoint and three "checkpoint" wafers. The checkpoint wafers, run at different levels than the experimental runs, were used to verify the models built using the data from the experiment. Because the process is proprietary by Texas Instru-

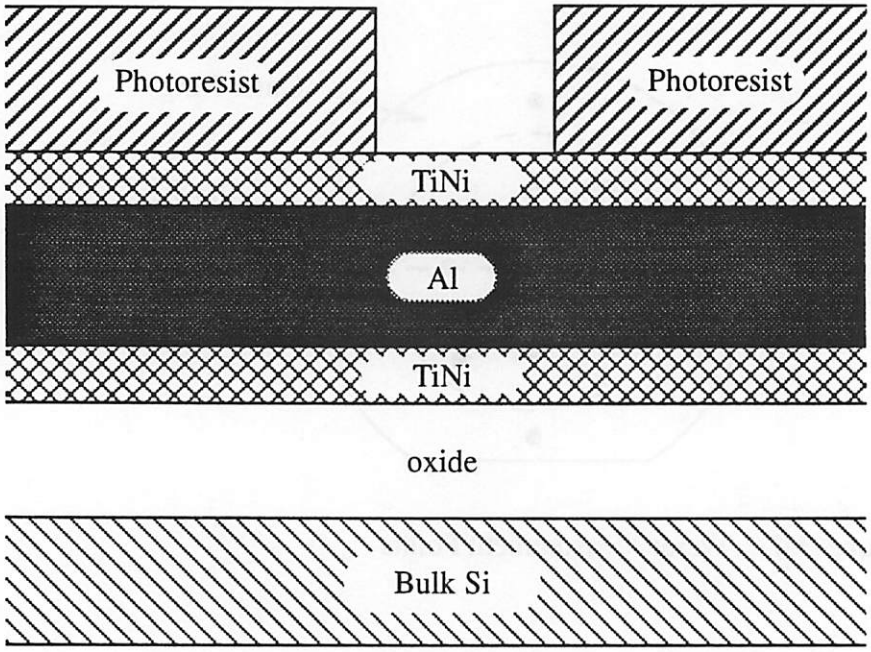


Figure 3.5 Test Structure for Lam TCP 9600 Experiments.

ments, the experimental design is listed in terms of percent change from nominal in Table 3.17. The three levels used in the experiment are shown in Table 3.16.

Table 3.16 Three Levels and Checkpoints in the TCP Static Experiment

Input Setting	-	+	checkpt 1	checkpt 2	checkpt 3
RF Top	- 40%	+ 40%	+ 14.3%	- 14%	- 21.4%
RF Bottom	- 20%	+ 20%	- 10%	+ 10%	- 10%
Cl_2	- 8.13%	+ 6.93&	- 4%	+ 4%	+ 6.67%
BCl_3	- 6.93%	+ 8.13%	+ 4%	- 4%	- 6.6%

Table 3.17 TCP Static Experiment

run #	RF Top	RF Bot	Cl_2	BCl_3	run #	RF Top	RF Bot	Cl_2	BCl_3
1	0	0	0	0	19	0	0	+	-
2	-	+	0	0	20	0	-	-	+
3	+	+	-	+	21	0	+	0	0
4	0	+	+	-	22	-	0	0	0
5	-	0	+	-	23	-	-	+	-
6	0	0	-	+	24	0	0	0	0
7	0	-	0	0	25	-	-	0	0
8	0	0	0	0	26	0	+	-	+
9	+	-	+	-	27	+	-	-	+
10	+	0	0	0	28	0	-	+	-
11	-	-	-	+	29	+	0	+	-
12	checkpoint 1				30	+	+	0	0
13	0	0	0	0	31	0	0	0	0
14	+	0	-	+	32	-	+	+	-
15	+	+	+	-	33	-	0	-	+
16	+	-	0	0	34	+	-	-	+
17	-	+	-	+	35	checkpoint 2			
18	0	0	0	0	36	checkpoint 3			

3.3.2 Dynamic Testing and Verification Experiments

Because we are interested in capturing trends in the data due to time, and the Static Experiment was conducted all in one day (hence the name "static"), we propose the following experiment for the second set of experiments. This Dynamic Experiment is

designed to produce information that can be used to map the input settings to the real-time data, while taking into account the time trends in the data.

Five inputs will be varied in the fractional factorial experiment: Top RF Power (A), Bottom RF Power (B), Cl_2 (C), BCl_3 (D), and Pressure (E). The first four parameters were varied in the first experiment, while pressure will be added in the second experiment. We propose a three-phase experiment run over a series of days or weeks. The first phase consists of half of the factorial design, as shown in Table 3.18. The fraction was determined from the blocking equation: block = ABCDE. The second phase consists of one or two lots of baseline data (nominal recipe). The first and second phases will be used to train the system, and is called the TCP Training Experiment. Finally, the third phase is the other half of the factorial design, as listed in Table 3.19. This third phase, called the TCP Verification Experiment, will be used to verify and check the validity of the models built using data collected during the Training Experiment. Note that both fractional factorial designs should be randomized before the actual experiment.

Table 3.18 TCP Dynamic Experiment: Training Phase I

Trial	RF Top	RF Bot	Cl_2	BCl_3	Pressure	block
1	+	+	+	+	+	+
4	+	+	+	-	-	+
6	+	+	-	+	-	+
7	+	+	-	-	+	+
10	+	-	+	+	-	+
11	+	-	+	-	+	+
13	+	-	-	+	+	+
16	+	-	-	-	-	+
18	-	+	+	+	-	+

Table 3.18 TCP Dynamic Experiment: Training Phase I

Trial	RF Top	RF Bot	Cl_2	BCl_3	Pressure	block
19	-	+	+	-	+	+
21	-	+	-	+	+	+
24	-	+	-	-	-	+
25	-	-	+	+	+	+
28	-	-	+	-	-	+
30	-	-	-	+	-	+
31	-	-	-	-	+	+

Table 3.19 TCP Dynamic Verification Experiment

Trial	RF Top	RF Bot	Cl_2	BCl_3	Pressure	block
2	+	+	+	+	-	-
3	+	+	+	-	+	-
5	+	+	-	+	+	-
8	+	+	-	-	-	-
9	+	-	+	+	+	-
12	+	-	+	-	-	-
14	+	-	-	+	-	-
15	+	-	-	-	+	-
17	-	+	+	+	+	-
20	-	+	+	-	-	-
22	-	+	-	+	-	-
23	-	+	-	-	+	-
26	-	-	+	+	-	-
27	-	-	+	-	+	-

Table 3.19 TCP Dynamic Verification Experiment

Trial	RF Top	RF Bot	Cl_2	BCl_3	Pressure	block
29	-	-	-	+	+	-
32	-	-	-	-	-	-

References for Chapter 3

[3.1] G. E. P. Box, N. R. Draper, *Empirical Model-Building and Response Surfaces*, Wiley, 1987.

[3.2] A. J. van Roosmalen, J. A. G. Baggerman, S. J. H. Brader, *Dry Etching for VLSI*, Plenum Press, 1991.

[3.3] J. P. McVittie, J. C. Rey, A. J. Bariya, M. M. IslamRaja, L. Y. Cheng, S. Ravi, K. C. Saraswat, "SPEEDIE: A Profile Simulator for Etching and Deposition," *SPIE: Advanced Techniques for Integrated Circuit Processing*, vol. 1392, 1990, pp. 126-138.

[3.4] V. Vahedi, R. A. Stewart, M. A. Lieberman, "Analytic Model of the Ion Angular Distribution in a Collisional Sheath," *J. Vac. Sci. Technol. A.*, vol. 11, no. 4, Jul/Aug 1993, pp. 1275-1257.

[3.5] M. A. Lieberman, R. A. Gottscho, "Design of High Density Plasma Sources for Materials Processing," UCB/ERL M93/3, Jan. 11, 1993.

[3.6] G. S. May, J. Huang, C. J. Spanos, "Statistical Experimental Design in Plasma Etch Modeling," *IEEE Trans. Semiconductor Manufacturing*, vol. 4, no. 2, May 1991, pp. 83-98.

Chapter 4

Fault Detection

4.1 Introduction

This chapter describes the module which detects equipment malfunctions in real-time. It has been shown by Guo and Spanos *et al.* that real-time tool signals can be used effectively to detect equipment malfunctions on a real-time basis [4.1][4.2]. Although effective for equipment fault detection in real-time, the original algorithm sometimes resulted in false alarms[1] at the start of a wafer. Moreover, the fault detection algorithm required training for each recipe on a given machine. While it may not pose a large problem for manufacturing houses which produce a few high volume products, training the module can become unwieldy for manufacturing houses with a large mix of products.

This thesis develops improvements to the original algorithm resulting in more robust fault detection with fewer false alarms. In addition, the algorithm has been expanded to

1. A false alarm occurs when the module generates an alarm indicating a problem when the process is actually in control.

accommodate several different recipes on the same machine. This chapter first gives an overview of the module and highlights the improvements made to the original fault detection algorithm, followed by a discussion of using multiple recipes.

4.2 Background and Motivation

To determine whether a machine is functioning properly, standard practice in industry includes building various statistical process control (SPC) charts based on monitor wafer output states or input settings. When the monitored parameter exceeds specified limits set by the process engineer (specification limits), an alarm is generated and a technician is summoned to diagnose and then correct the problem. Examples of measured wafer states include etch rate and wafer uniformity. To determine whether or not the wafer states of interest are within the specification limits, monitor wafers are usually run and measured on a regular basis, perhaps at the start of each shift and after machine recipe changes (change of the input settings on the machine). Unfortunately, machine problems which occur between monitor wafers are undetected until the next monitor wafer is run. This delay in fault detection can result in considerable scrap produced by the equipment.

In addition to monitor wafers, signals corresponding to the input settings may be monitored for every wafer processed in the equipment. Examples of monitored signals include the chamber pressure or gas flows. Although control charts based on these equipment signals can detect faulty wafers during production, this method also suffers from some major drawbacks. One serious problem is that the monitored signals may not be issuing the correct information. For example if a mass flow controller (mfc) is miscalibrated, although the reading on the mfc may be within specifications, the actual flow rate may be outside of the specification limits.

Another disadvantage of using this system of plotting individual control charts for each monitored signal of interest is that as the number of monitored signals grow, so does the number of control charts that the operator must monitor. Eventually, there may be too many charts for an operator to realistically monitor. A much more serious problem arises when the variables plotted in the control charts are correlated. It can be shown that the false alarm rate escalates quickly when several signals are correlated [4.3].

The Fault Detection Module presented in this thesis eliminates the problems outlined above. Instead of relying on monitor wafers or signals based on the input settings to detect equipment malfunctions, the module uses real-time tool data automatically collected from the machine. As described in section 2.3, the real-time signals monitored consist of both electrical and mechanical signals which reflect the actual state of the machine. For example, instead of collecting signals from a gas mfc directly, the fault detection module uses the throttle position and other signals to glean information about the gas flow. The goal is to monitor those signals which are most sensitive to the actual equipment state. Through extensive experimentation and analysis described in Chapter 2, the sets of signals (among those collected by the monitoring systems used in this work) most useful for fault detection for plasma etchers are listed in section 2.3.

One may be tempted to simply use the standard SPC chart to monitor the real-time data. Applying standard control charts to the real-time data, however, is not a viable method. Although the real-time signals contain information about the equipment state, control charts can not be applied effectively to real-time tool data. Because the real-time tool signals are collected at either 1 or 2 Hz (depending on which monitor is used) time series patterns are observed both within each wafer and across several wafers due to controller adjustments and equipment aging. These signals are highly auto- and cross-correlated. In addition, the correlation structure and the mean value for a given signal may also

vary with time, making the series non-stationary. Thus, the data are not identically, independently, normally distributed (IIND), and can not be used directly in a traditional control chart such as a Shewhart or $\bar{X}$ - R chart [4.3].

Figure 4.1 shows an example of a standard Shewhart chart applied to an endpoint trace for eight normal production wafers. The data is filtered as described in section 2.4 so that only data from the main etch step are included. Note that the endpoint trace for each wafer is highly auto-correlated, as seen by the rising trend for the first few seconds of each wafer etch, followed by a steep downward trend. These trends, which occur naturally in the data, result in alarms when the Western Electric Company (WECO) rules for SPC charts are applied [4.4]. Following the WECO rules, the process is considered to be out-of-control if one or more of the following occur:

- One point plots outside of the 3-sigma control limits.
- Two of three consecutive points plot beyond the 2-sigma limits.
- Four of five consecutive points plot at a distance of 1-sigma or beyond from the center line.
- Eight consecutive points plot on one side of the center line.

In addition, Figure 4.1 shows that the overall mean of the endpoint trace across several wafers changes with time, causing the endpoint trace to dip below the lower control limit (LCL), resulting in alarms. Since the endpoint data is from production wafers in statistical control, the autocorrelation and varying mean will result in false alarms.

To make matters worse, applying standard control charts to real-time data also results in a high missed alarm[1] rate. Using the same example in Figure 4.1, a change in the trend or shape of the endpoint trace will not be detected using the control limits as shown. Thus,

1. A missed alarm results when an alarm is not generated when the process is out-of-control.

to lower the false alarm rate the control limits are widened, which increases the missed alarm rate.

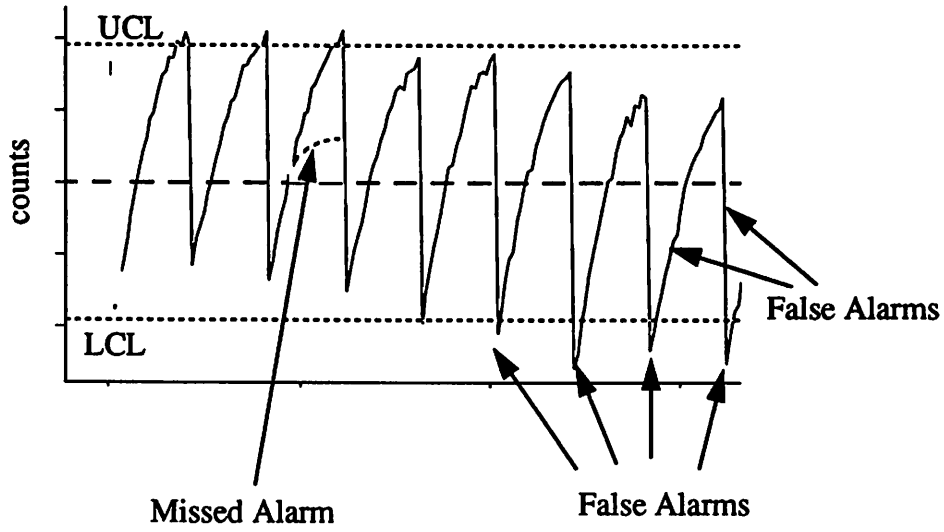


Figure 4.1 Shewhart control chart applied directly to endpoint trace.

4.3 Fault Detection Algorithm

Given the above problems with applying control charts directly to real-time data, a different approach is taken by the Fault Detection Module to utilize the information found in the real-time data to perform "real-time SPC." The module first learns the shape of the in-control real-time data, and later detects deviations from this shape. More specifically, time series models are utilized to analyze the real-time signals available from manufacturing equipment. The models built from data collected while the machine is in-control establish the baseline behavior of the machine, and are called baseline models. When subsequent production wafers are processed in the machine, the fault detection module detects deviations from the baseline models in the new signals, and generates alarms. The following sections describe the algorithm in more detail, beginning with time series models.

4.3.1 Baseline Behavior Modeling

4.3.1.1 Time Series Models

The first step in the algorithm is to model each signal using a time series model, which accounts for the expected patterns in the data. Once these patterns (whose presence does not indicate a malfunction) are filtered from the signal, deviations can be detected in the filtered signals, suggesting that a malfunction, or equipment fault, has occurred. The time series model captures the dependencies among sequential readings of the same process variable. Dependencies within readings collected over time can be described by univariate time series models such as ARIMA(p, d, q) models, where p is the auto-regressive order, d is the integration order, and q is the moving average order. The form of the equation for a non-stationary[1] time series x_t with autoregressive parameters ϕ_k and moving average parameters θ_k is [4.5]

$$w_t = -\sum_{k=1}^{p} \phi_k w_{t-k} + \sum_{k=0}^{q} \theta_k a_{t-k} \tag{4.1}$$

where $\theta_o = 1$, $|\phi_1| < 1$, the error $a_t \sim N\left(0, \sigma^2\right)$, and w_t are the differenced data

$$w_t = \nabla^d x_t \tag{4.2}$$

where ∇^d is the d^{th} order of differencing operator, and

$$\nabla^1 x_t \equiv x_t - x_{t-1},\ \nabla^2 x_t \equiv \nabla^1 x_t - \nabla^1 x_{t-1} = x_t - 2x_{t-1} + x_{t-2}, \ldots. \tag{4.3}$$

The assumption behind the univariate analysis is that a significant portion of a real-time signal's behavior can be explained by using past observations of the signal. A more thorough explanation of time series models is given in [4.5][4.6][4.7] and [4.8].

---

1. A stationary series has a constant mean, variance, and autocorrelation through time. A non-stationary series can often be made stationary by differencing the data [4.5][4.7].

ARIMA(p, d, q) models can be derived from the collected data when the process is under statistical control; in this way the models describe the baseline behavior of the process. Once developed, the models are used with current readings to forecast each new value. The difference between the forecasted value and the actual value of the signal from the production wafers is the forecasting error, or residual. When the equipment is in statistical control, the residuals are by definition IIND variables. These IIND residuals can then be plotted in standard SPC charts to perform "real-time SPC." An example of a baseline model and in-control production data is shown in Figure 4.2.

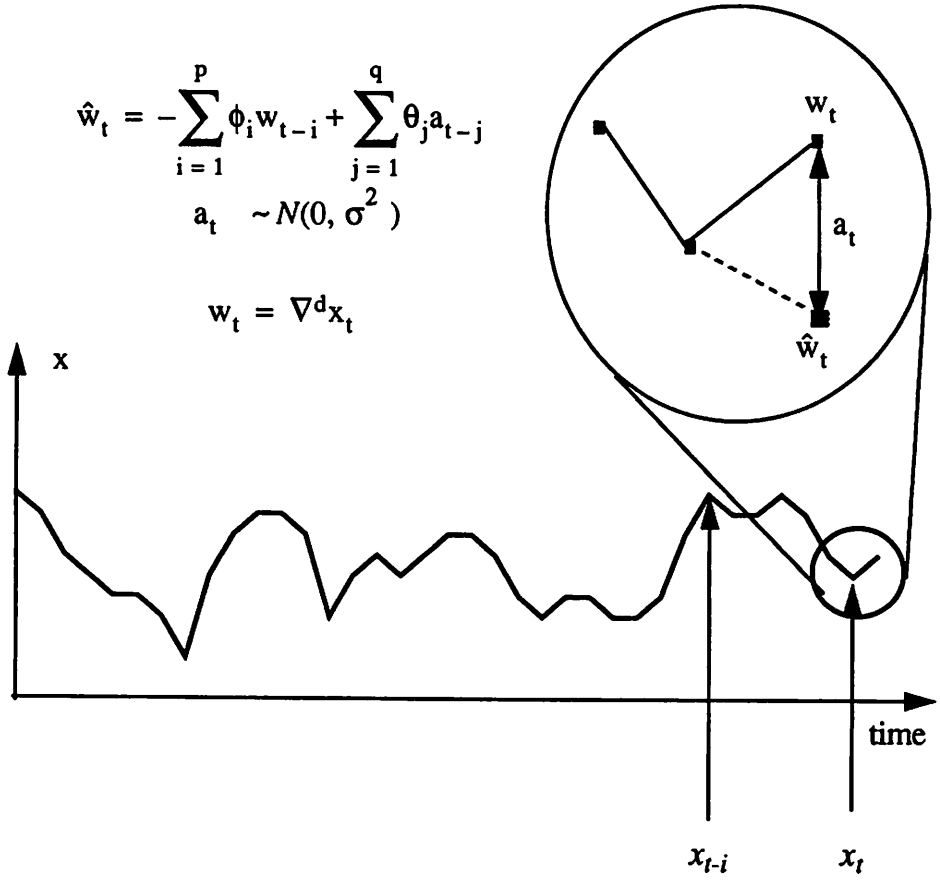


Figure 4.2 ARIMA(p, d, q) model: The signals from the production wafers w_t are compared to the model of the baseline wafers $\hat{w}_t$, resulting in residuals a_t which can be plotted in a standard SPC chart.

4.3.1.2 Decomposition of Real-Time Data

The algorithm presented in [4.2] builds one seasonal ARIMA (SARIMA) model for each sensor variable. As previously mentioned, a major disadvantage of this algorithm is that false alarms often occur at the start of a wafer. While these false alarms can be anticipated and ignored, the new algorithm addresses this problem more formally. First, SARIMA models are not appropriate to model the real-time data, because as described in section 2.2.2, the pre-filtered wafer signals from the main etch step are concatenated. This concatenation means the data do not form a natural continuous stream. One assumption behind the SARIMA model is that the variance and the mean of the filtered residuals is the same regardless of the season. Since the discontinuity violates this assumption, the idea of seasons is eliminated in the new algorithm.

The most significant change in the algorithm is the decomposition of the real-time signals from each sensor into long-term and short-term components before modeling [4.10][4.11]. This decomposition is necessary because each component describes a different behavior of the process. An example of signal decomposition of the impedance signal for several wafers is shown in Figure 4.3. The long-term component, comprised of the average value of the signal for each wafer, models the overall trend across a number of wafers. On the other hand, the smaller deviations within each wafer create the short-term component, which captures the short-term patterns during the processing of each wafer. Most importantly, the variation of the long-term component is much larger than that of the short-term component, illustrating the point that the short-term components are more sensitive to faster equipment fluctuations, while the long-term components reflect longer duration changes in overall equipment state. This decomposition of the signals into components with drastically different variances is the primary reason the false alarm rate has been decreased. To simplify later calculations, the short-term components are demeaned

by wafer, while the long-term components are demeaned by an average of the baseline wafers after the decomposition.

Notice that the short-term component for each wafer in Figure 4.3 roughly follows a downward trend. This trend, modeled by the integrative part of the ARIMA model, is captured for each wafer so that deviations from this trend will be detected. Deviations in each of the components reflect different changes in equipment state. For example, a shift in RF power that lasts the duration of the wafer etch will be seen as a shift in the long-term signal. A short spike in RF power, however, will be exhibited in the short-term signals. As another example, a dirty film on the wafer results in an alarm by the short-term signals but not by the long-term signal. Because the decomposition allows us to model two different types of faults, the resulting algorithm is more robust than the original method, gives significantly fewer false alarms, and generates residuals that are much more suited for diagnosis.

4.3.2 Monitoring Production Wafers

Once baseline behavior has been established, production wafers can be run through the machine. As in the training case, the real-time signals from the production wafers are decomposed into long- and short-term components. In single wafer processing equipment, these components represent the wafer-to-wafer averages and the within-wafer signal trends, respectively. Each component is then filtered using the respective baseline time series model. The residuals $\mathbf{x}$ (the difference between the actual and forecasted baseline values) for each component are then combined using the multivariate Hotelling's T^2 statistic into a single score[1]:

$$T^2 = n(\mathbf{x}-\mathbf{0})'\mathbf{S}^{-1}(\mathbf{x}-\mathbf{0})$$

1. Bold face upper case letters denote matrices. Lower case bold face letters and Greek letters with an underscore ($\underline{\ }$) denote column vectors. Scalars are denoted by lowercase letters. Transpose is denoted by (').

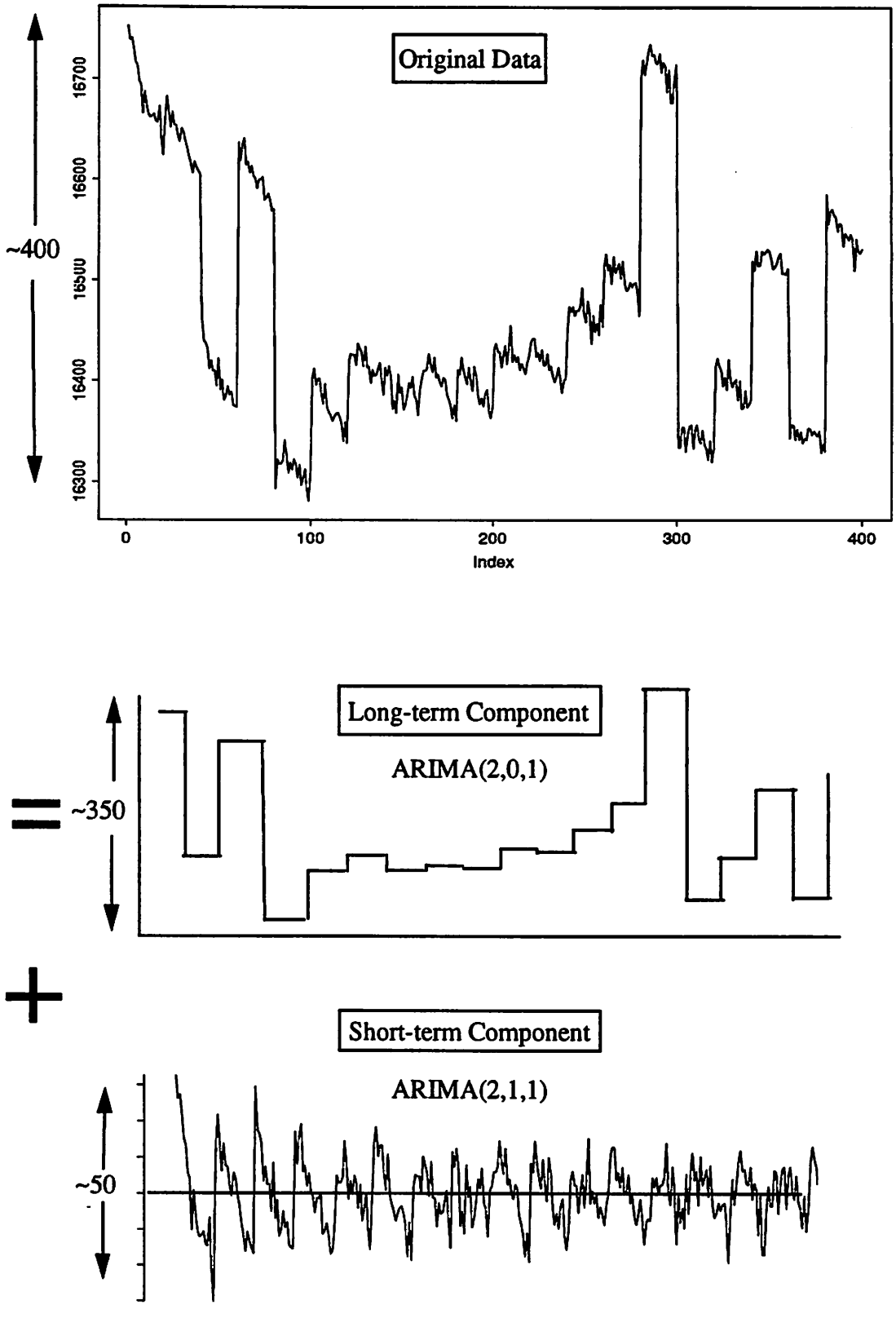


Figure 4.3 Real-time Signal Decomposition for the Impedance Signal

where n is the total number of observations and S is the estimated variance-covariance matrix of the residuals used to build the original baseline models.

The scores are graphically displayed in the resulting double T^2 control chart. The use of the Hotelling's T^2 statistic reduces the problems associated with several control charts of correlated signals resulting in a high false alarm rate [4.3]. The resulting Hotelling's T^2 scores for each component are plotted in a one-sided SPC chart. The upper control limits (UCLs) are scaled so that both sets of scores can share the UCL on the same control chart. Data points corresponding to run-time faults have residuals which cause the Hotelling's T^2 statistic to be significantly different from zero. One set of scores, obtained from the short-term components, detects faults during the process time of each of the wafers, while the second set of scores, obtained from the long-term components, detects faults by looking at violations in trends across several wafers.

If no equipment faults are detected, normal operation of the machine continues. When a malfunction is detected, the diagnostic routine is triggered, and an alarm is generated to alert the operator[1]. Diagnosis currently uses the long-term residuals (the difference between the actual real-time signal averages for that wafer and the time series model predictions for the signal averages) as a signature of the specific equipment malfunction [4.9]. An overview of the real-time SPC data analysis flow is shown in Figure 4.4.

This new algorithm has been implemented in RTSPC, a software package which includes automated model generation [4.10], data filtering, and a novel double T^2 graphical control chart for the display of alarm conditions [4.11]. RTSPC interfaces with a work-cell controller and can serve as a platform for future real-time process control.

1. In the examples shown, the relevant data was manually transferred to the diagnostic module. An automated link is currently under development.

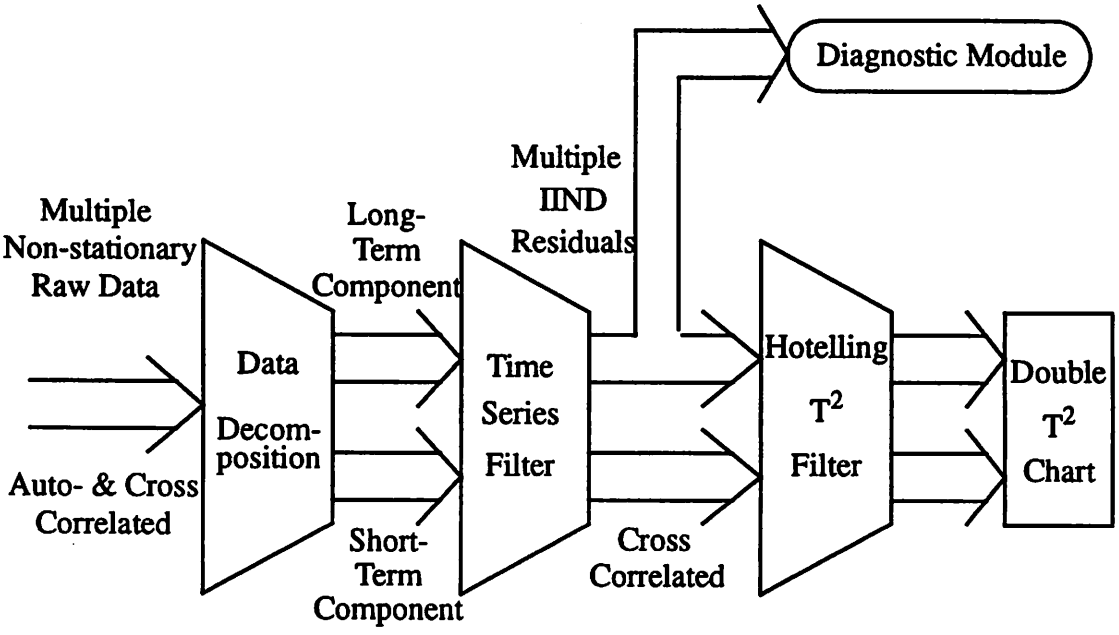


Figure 4.4 Real-Time SPC Data Flow: First, the raw data is decomposed into the long-term and short-term components. They are each filtered using time series models. The resulting residuals are then applied to a Hotelling's T^2 filter, and the scores for each set of components are plotted in the double T^2 chart. If a fault is detected, the long-term component is siphoned to the Diagnostic Module.

4.4 Fault Detection Example

To demonstrate the capability of RTSPC, several experiments were conducted in the Berkeley Microfabrication Laboratory. As described in the previous chapters, the equipment chosen for the experiment was the Lam Rainbow 4400 plasma etcher. The following sections show examples of both baseline and production processing.

4.4.1 Baseline Processing and Model Generation

To use RTSPC on a specific process, a set of baseline wafers must first be processed to build the time series models needed for data filtering. During the processing of these

wafers, it is essential that the machine be operating in statistical process control. In this example 11 baseline wafers were first processed on the Rainbow 4400 using the given recipe. The real-time data from five signals (RF impedance, RF phase, endpoint, coil and tune vane position) were selected to generate a model set using the automatic model generation routine.

The results of the baseline model generation are shown in Figure 4.5, which displays the double T^2 chart for the data used to generate the model. The user may also view the signals and residuals resulting from the baseline model. Note that in double T^2 chart the long-term signal is never out of control, although the short-term signals show some points close to the control limit. If the baseline contains alarms, the user may choose to eliminate the wafers showing alarms and rebuild the model. Note, however, that for an expected false alarm rate of 0.05, one would expect on average 5 points indicating alarms out of every 100 points. Therefore, it is normal if on occasion the baseline data indicates alarms, even when the process is in control.

4.4.2 Real-Time SPC During Processing

Once the time series models for the baseline of this process are created, the RTSPC software generates real-time alarms in the case of misprocessed wafers. To demonstrate the alarm generation capability of RTSPC, an additional 15 wafers were processed with the baseline recipe, along with 3 runs with intentional faults. The same sensor data used in the baseline models were collected and analyzed by RTSPC.

The results for the additional wafers are shown in Figure 4.6. All but the sixth, twelfth, and eighteenth wafers were processed when the equipment was in control, and no long- or short-term component alarms for these wafers were signaled by RTSPC. The sixth wafer was processed with a 10% decrease from the baseline value of pressure. The gas flow ratio of the twelfth wafer was decreased by 10% from the baseline value. The power was

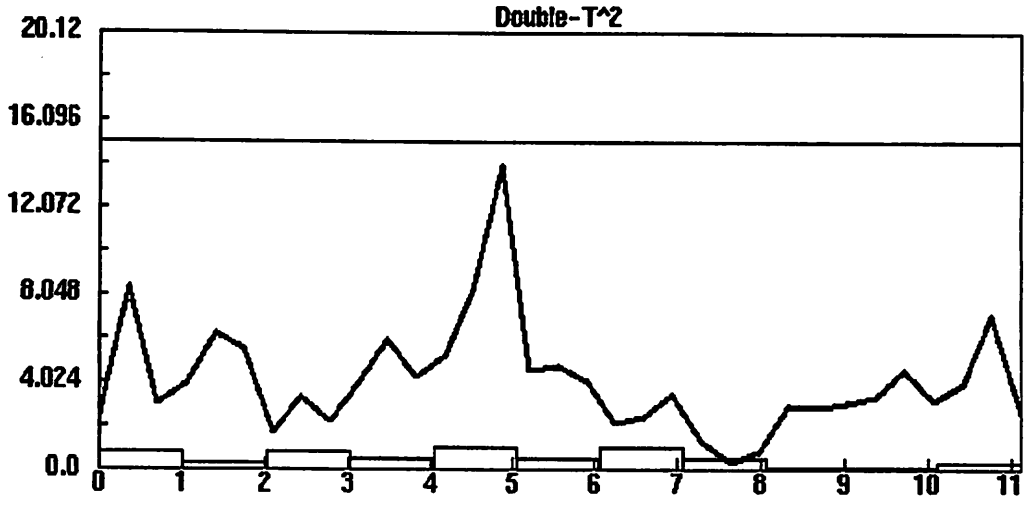


Figure 4.5 Baseline Double T^2 Chart. Shown in the one-sided control chart are the results from both the short-term component and the long-term component, so that information can be obtained on both a within wafer and wafer-to-wafer basis. The control limit is scaled so that data from both components can be plotted on the same chart.

increased by 10% from the baseline value while the eighteenth wafer was processed. The injected faults simulate a fault in the equipment such as a problem with the chamber pumping system, miscalibration of a mass flow controller, or a malfunction in the RF matching network. All three wafers with injected faults resulted in long-term component alarms. In the current implementation of the RTSPC software, the long-term component alarm changes color (from green to red on the UNIX console), giving clear visual indication that a malfunction occurred.

Although not shown, cases exist when the short-term components alone generates alarms. An example of such alarms occur for wafers with dirty films. These alarms imply that although the mean values of the real-time signals were in control, the time series pattern of the signals during processing was altered as the equipment compensated for the

change in film quality. These alarms further show the sensitivity of the RTSPC algorithms and the importance of the real-time signal decomposition.

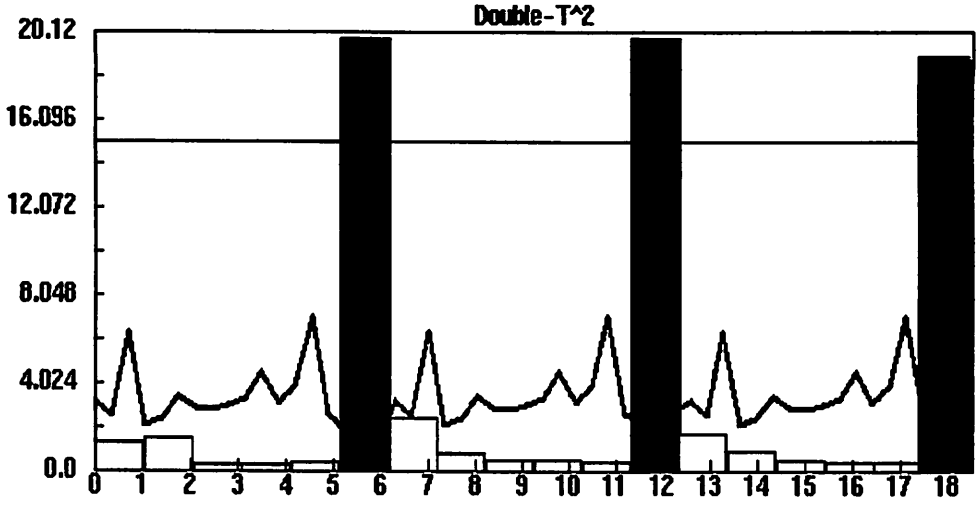


Figure 4.6 Graphical Display of Production Double T^2 Control Chart. The sixth, twelfth, and eighteenth wafers trigger long-term component alarms. The gas flow ratio of the twelfth wafer was decreased by 10% from the baseline value. The power was increased by 10% from the baseline value while the eighteenth wafer was processed.

The RTSPC software can display the real-time signals and residuals to give the operator additional insight into the cause of the alarms. Figure 4.7 shows the signal for the RF coil position and its associated long- and short-term residuals, called wafer-to-wafer and within-wafer residuals, respectively. The residual plots beneath the signal clearly show if components of the signal caused the alarm. Note the large shifts in the coil position during the processing of all three faults, as the equipment compensates for the miscalibrations. This shift in the means is clearly seen in the wafer-to-wafer residual plot, which shows large residuals in the cases of wafers 6, 12, and 18. When the machine is in control, the wafer-to-wafer residuals do not exceed the 3-sigma upper and lower control limits. In this

example, the within-wafer residuals are in control for all wafers. Each modeled component can be viewed in the same manner. In the case of an alarm, the software's interactive capability allows the user to selectively view any of the signals or residuals to aid in diagnosis of the malfunction.

The above example illustrates the important point that several faults map onto changes in each real-time signal. In the example, all three faults resulted in an positive shift in the mean of coil position. To distinguish the faults, other signals must be examined. Therefore, in addition to alarm generation and the display of individual signals, there is a need for a diagnosis module which can interpret the signatures of the faults to classify faults to specific equipment problems. Work on the diagnosis algorithms is the subject of the following chapter.

4.5 Multiple Recipes

The fault detection algorithm described in the previous section is extremely sensitive, catching faults at 5% from nominal values. The algorithm has been successfully applied to different types of etchers, including both parallel plate and TCP machines. Various faults have been detected, such as having the improper wafer in the chamber, faulty mfc's, miscalibrated electrode gap spacing, spikes in the RF power, and changes in the chamber pressure. The limitation of the algorithm is that the baseline behavior of the real-time signals must be learned for each set of input settings, called a recipe, used on each machine. Furthermore, processes with different loading from different mask patterns and exposed surface on the wafers also require individual training runs.

This section proposes an algorithm to train the fault detection module to recognize several different recipes without having to train each one separately. The main idea is to use both time series and linear regression models. As before, the time series models cap-

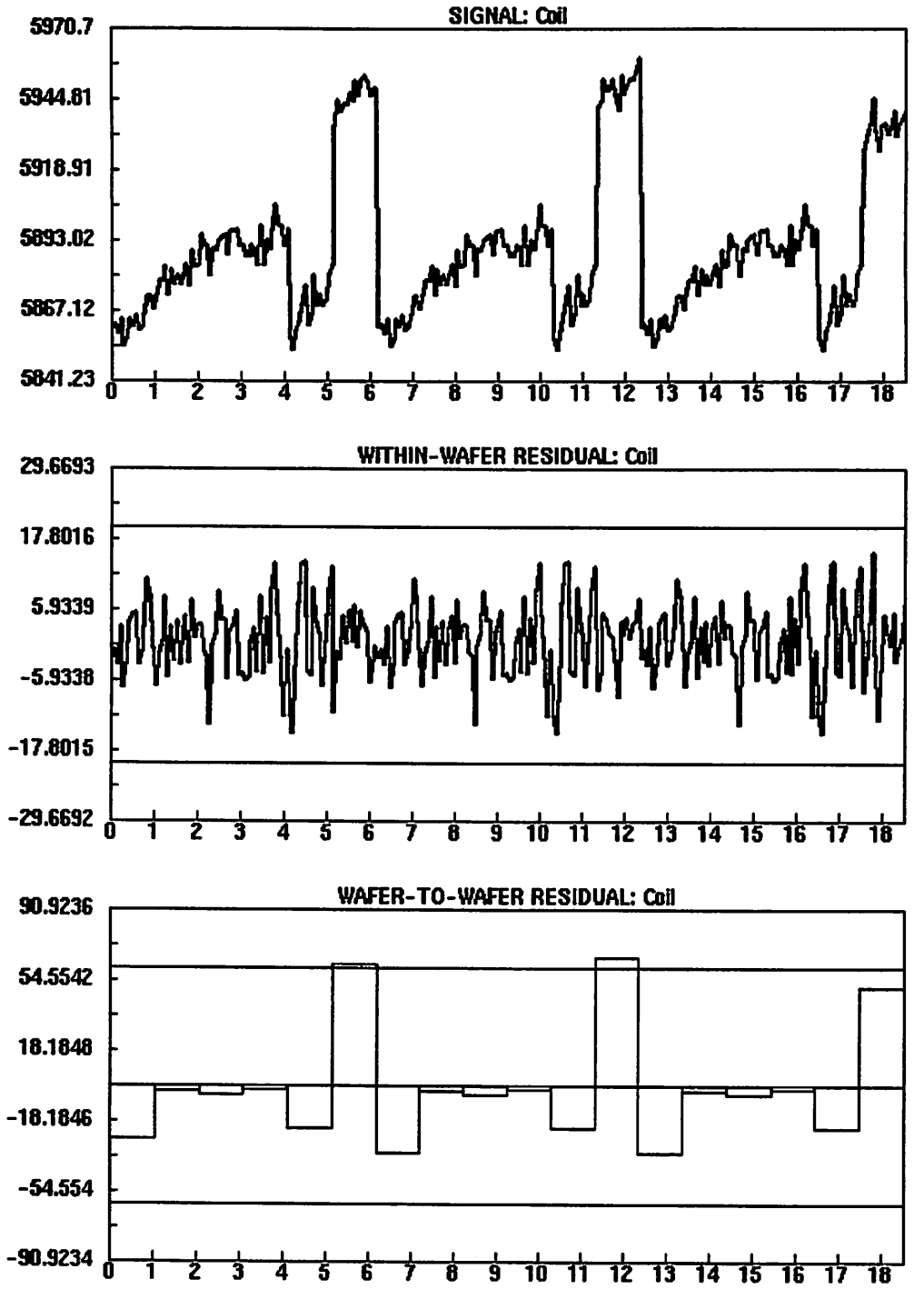


Figure 4.7 Real-Time Signal and Residual Plots for Coil Position

ture the time domain trends seen in the data. Added to these models are the linear regression models, which predict the effect of recipe changes.

The general class of models employed are called ARIMAX models, which is an extension of ARIMA models. The "X" stands for "exogenous," which simply means the model now contains additional explanatory variables. The next section briefly describes ARIMAX models. A more thorough discussion is given in [4.12][4.13][4.14].

4.5.1 ARIMAX models

ARIMAX models, also known as Transfer Function Models, forecast a time series using more than one time series from other variables, thus introducing explicitly the relationship among the signals. The overall ARIMAX model for a stationary series y_t based on the stationary time series x_t has the following form [4.13][4.14]:

$$y_t = \frac{C\omega(B)}{\delta(B)} x_{t-b} + \frac{\theta(B)}{\phi(B)} a_t \tag{4.4}$$

where

- B is the *backshift* operator defined as: $B^k x_t = x_{t-k}$ for integers k
- C is an unknown scale parameter
- the delay b is the number of time periods before x_t begins to influence y_t
- $\omega(B) = \omega_o - \omega_1 B - \ldots - \omega_l B^l$ is the x_t operator of order l where l represents the number of past x_t values which influence y_t
- $\delta(B) = \delta_o - \delta_1 B - \ldots - \delta_r B^r$ is the y_t operator of order r where r represents the number of past y_t values which influence itself
- $\theta(B) = \theta_o - \theta_1 B - \ldots - \theta_q B^q$ where q is the number of moving average terms in the ARIMA(p, d, q) model for the error component
- $\phi(B) = \phi_o - \phi_1 B - \ldots - \phi_p B^p$ where p is the number of autoregressive terms in the ARIMA(p, d, q) model for the error component
- and error $a_t \sim N\left(0, \sigma^2\right)$.

The first term in the model accounts for the cross-correlation between series x_t and output y_t, while the second term models the error not taken into account by the correlation. On other words, the first term accounts for the systematic portion of the model based on x_t while the second term is an ARIMA(p,d,q) model for the error. If *m* parameters are included in the model, Equation (4.4) can be easily extended to

$$y_t = \sum_{i=1}^{m} \frac{C_i \omega_i(B)}{\delta_i(B)} x_{i,t-b} + \frac{\theta(B)}{\phi(B)} a_t. \tag{4.5}$$

Figure 4.5 shows a schematic of the above transfer function model [4.13]. The top section of the figure shows the transfer function which determines the influence of the explanatory variables $x_{1,t}, x_{2,t}, \ldots, x_{i,t}, \ldots, x_{m,t}$ on the dependent variable y_t. The lower section shows the univariate model for the noise term, modeled with a standard univariate ARIMA(p, d, q) model.

In this application, a simplified version of the above equation was employed to model both the time series and recipe changes. First, the input settings from an experimental design were fitted in a linear regression model to model each real-time parameter. Next, the residuals, which contain a time component, are fitted using ARIMA(p, d, q) models. This is equivalent to setting the ratio $\frac{\omega_i(B)}{\delta_i(B)} = 1$ in the above equation and setting C_i to the coefficients to the linear regression model.

4.5.2 Example of Multiple Recipes

The experiment conducted to evaluate this theory was performed at Texas Instruments on a metal etcher, as described in section 3.3. Ten real-time signals were collected and four input settings were varied during the experiment. The eleven real-time signals are: RF Tune Vane Position, RF Load Coil Position, Line Impedance, RF Phase Error, DC Bias, TCP Tune Vane Capacitor Position, TCP Load Capacitor Position, TCP Line Impedance, TCP Phase Error, Endpoint, and RF Bias. They are described in greater detail in section 2.3.2. The input settings were: RF power of the top coil, RF power of the bottom

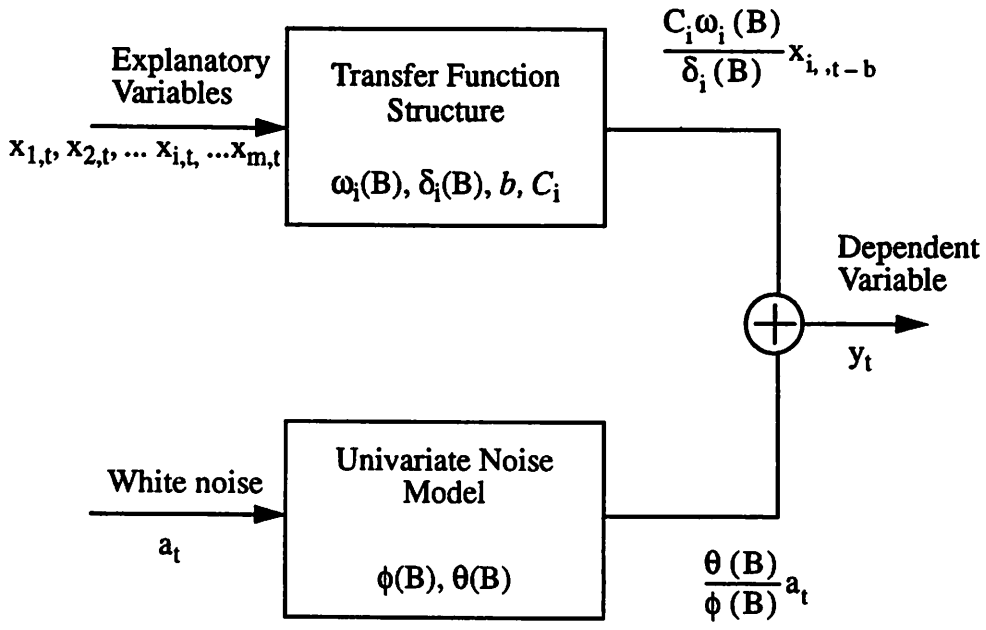


Figure 4.8 Transfer Function Model. The top section of the figure shows the transfer function which determines the influence of the explanatory variables $x_{1,t}, x_{2,t}, \ldots, x_{i,t}, \ldots, x_{m,t}$ on the dependent variable y_t. The lower section shows the univariate model for the noise term, modeled with a standard univariate ARIMA model [4.13].

electrode, Cl_2 flow, and BCl_3 flow. The data from the 32 runs were used to build regression models for each long-term component for each signal. The statistical software package S-PLUS was used to build both the regression and time series models .

All four input settings and their corresponding two-way interactions, for a total of 10 parameters, were used as input to build the models, with the exception of TCP Load Capacitor Position, which required all the main squared terms in addition. Because the input settings having different units, which could potentially bias the models, the input data were scaled so that each parameter has a mean of 0 and variance of 1. Principal com-

ponent regression (PCR) models were built for each long-term component. The first seven principal components (PC's) explain 99% of the variation in the data. Models were chosen so that the coefficients of each term in the model is significant at the 0.05 level. PCR is described in more detail in section 6.3.3. The results of the regression models are summarized in the following ANOVA tables. The ANOVA tables show the degrees of freedom (d.f.), the sum of squares (SS), and mean sum of squares (MS) for the regression, the residuals from the regression, and the residuals from the ARIMAX models. The test statistics for the regression models, which follow F-distributions, are calculated as the regression MS divided by the residual MS. The corresponding P-values give a measure of the model significance. Also listed is the adjusted R^2 statistic for the regression models, which takes into account the number of terms used in the model.

Note that models for the long-term components of RF Line Impedance and RF Phase Error are poor, in terms of both the F and the adjusted R^2 statistics. In addition, significant models for the long-term components of TCP Line Impedance and TCP Phase Error could not be found, and thus are not listed. This is because for these signals the range of the centerpoint data is large compared with the experimental runs, indicating that the input settings do not greatly affect these real-time signals.

Table 4.1 ANOVA Table for TCP Tune Vane Capacitor Position

Source	d.f.	SS	MS	F	P-value
Regression	5	9.15e7	1.83e7	2505	0
Residual	30	2.19e5	7.30e3		
adj. $R^2 = 0.997$					

Table 4.2 ANOVA Table for TCP Load Capacitor Position

Source	d.f.	SS	MS	F	P-value
Regression	3	6.19e6	2.06e6	48.62	5.07e-12
Residual	32	1.36e6	4.246e4		
ARIMAX	28	8.74e5	3.13e4	1.36	0.206
adj. R^2= 0.804					

Table 4.3 ANOVA Table for Endpoint

Source	d.f.	SS	MS	F	P-value
Regression	5	5.92e8	1.18e7	552.4	0
Residual	32	6.43e6	2.14e5		
ARIMAX	28	5.17e6	1.85e5	1.16	0.348
adj. R^2 = 0.988					

Table 4.4 ANOVA Table for RF Bias

Source	d.f.	SS	MS	F	P-value
Regression	6	1.99e4	3.31e3	333.9	0
Residual	29	287.7	9.92		
ARIMAX	24	186	7.75	1.28	0.271
adj. R^2 = 0.983					

Table 4.5 ANOVA Table for RF Load Coil Position

Source	d.f.	SS	MS	F	P-value
Regression	5	3.42e6	6.83e5	362.6	0
Residual	30	5.65e4	1884		
ARIMAX	29	5.40e4	1863	1.01	0.490
adj. R^2 = 0.980					

Once the regression models have been determined, the residuals are modeled using

Table 4.6 ANOVA Table for DC Bias

Source	d.f.	SS	MS	F	P-value
Regression	5	1.98e4	3.96e3	358.5	0
Residual	30	331.1	11.0		
ARIMAX	26	249	9.58	1.15	0.361
adj. $R^2 = 0.981$					

Table 4.7 ANOVA Table for RF Line Impedance

Source	d.f.	SS	MS	F	P-value
Regression	1	1.31e5	1.31e5	12.39	0.001253
Residual	34	3.60e5	1.06e4		
ARIMAX	32	3.15e5	9.84e3	1.08	0.415
adj. $R^2 = 0.207$					

Table 4.8 ANOVA Table for RF Phase Error

Source	d.f.	SS	MS	F	P-value
Regression	2	2.09e6	1.05e6	7.53	0.00202
Residual	33	4.58e6	1.39e5		
adj. $R^2 = 0.281$					

Table 4.9 ANOVA Table for RF Tune Vane Position

Source	d.f.	SS	MS	F	P-value
Regression	6	2.44e6	4.07e5	1421	0
Residual	29	8.32e4	287		
ARIMAX	25	6.65e3	266	1.08	0.462
adj. $R^2 = 0.996$					

ARIMA models, as described in section 4.3.1.1. The ARIMA model orders for the residuals which could be modeled are listed in Table 4.10. When combined with the regression

models, the time series models result in smaller absolute MS values for the majority of the signals. Figure 4.6 shows the predicted vs. actual plot for the TCP Load Capacitor Position. Both the original regression model and the new ARIMAX model are plotted. The figure shows that the ARIMAX model results in slightly better models.

Table 4.10 ARIMA(p, d, q) Models for the Residuals

Real-Time Signal	ARIMA(p, d, q)
TCP Load Capacitor Position	ARIMA(2, 0, 2)
RF Tune Vane Position	ARIMA(2, 0, 2)
RF DC Bias	ARIMA(2, 1, 1)
RF Load Coil Position	ARIMA(1, 0, 0)
RF Line Impedance	ARIMA(1, 0, 1)
Endpoint	ARIMA(1, 0, 1)
RF Bias	ARIMA(3, 1, 1)

The improvement in the models, unfortunately, is not statistically significant when tested with the F-statistic, as shown in Table 4.1 through Table 4.9. The second F-statistic in the ANOVA tables compares the regression and ARIMAX models, and is calculated as the ratio of the residual MS to the ARIMAX residual MS. The statistically insignificant improvement in the models may be due to the fact that the experimental runs were conducted in consecutive order all in the same day. Thus, little time effect was apparently captured in the data. Although the ARIMA models add improvement to the regression models, the original models are already very well-fitted so the additional improvement to the models is not as drastic as it may be when a larger time dependence appears in the data. When the equipment is operational on the factory floor, the time component in the data will be more significant than seen in this experiment. Thus, because significant time series models can be built from the regression residuals and as a result the absolute resid-

ual MS values are smaller for ARIMAX models, further study of the algorithm is warranted.

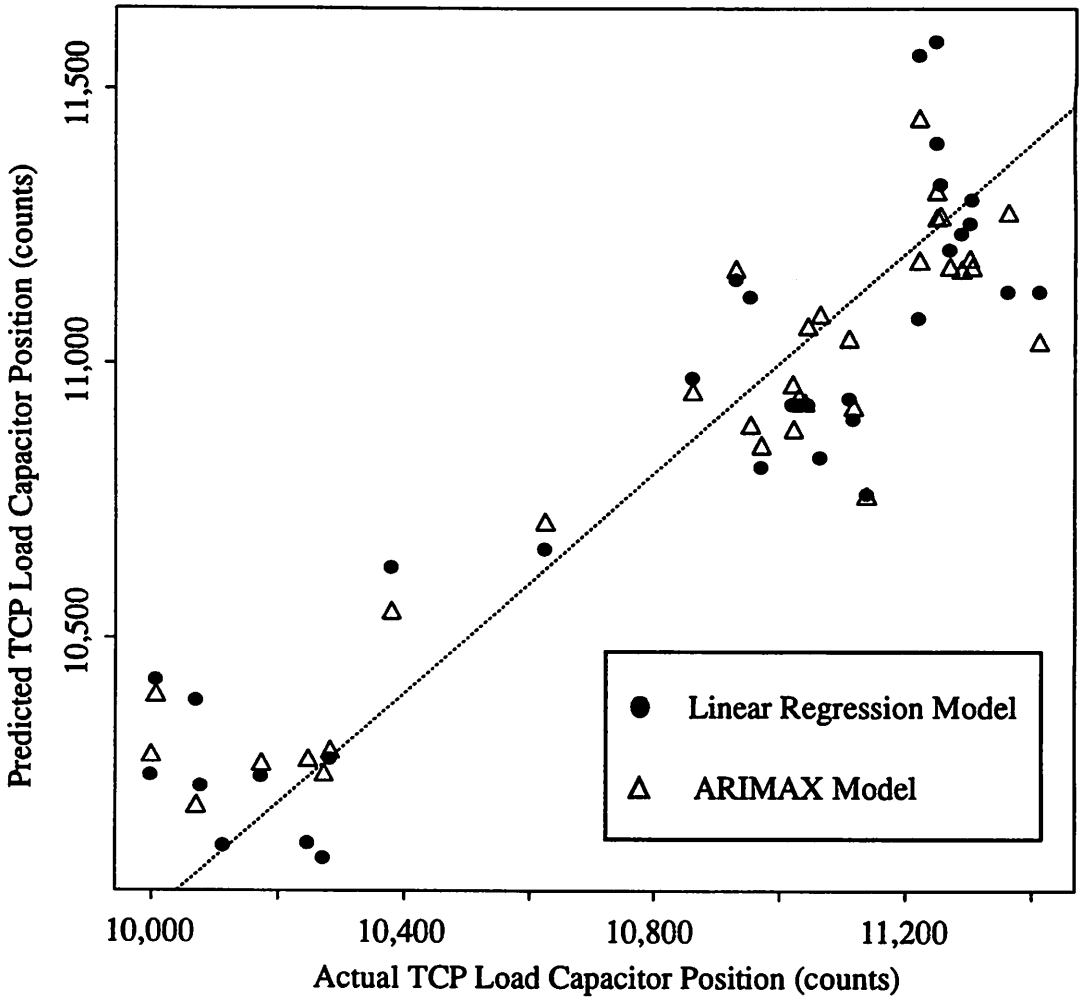


Figure 4.9 Predicted vs. Actual plot of TCP Load Capacitor Position. Both results from (1) standard principal component regression and (2) ARIMAX models are shown.

4.5.3 Future Work on the Multiple Recipe Algorithm

To simulate more realistic use of the equipment, and thus drifts in the equipment due to time, we proposed a second experiment which will be conducted at Texas Instruments

over a span of several weeks. In addition to adding a time element to the experiment, we make sure the experiment itself has not overshadowed the actual time patterns in the data. Thus, the experiment, described in detail in section 3.3, attempts to separate the experimental design and the time patterns in the data. More specifically, the experiment consists of a fractional factorial design from which the linear regression models for the long-term component signals can be determined, followed by a series of baseline runs from which the time series models can be obtained for each signal. The third phase of the experiment consists of another fractional factorial design which can be used to test the algorithm.

4.6 Fault Detection Module Summary

The fault detection module is powerful in that it uses the non-invasive real-time signals automatically collected from the equipment during run-time to detect equipment malfunctions. These faults include any abnormal behavior of the machine ranging from miscalibrated mfc's to loading the incorrect cassette of wafers into the machine.

In this chapter, we have demonstrated an improved algorithm for use in real-time SPC applications. This algorithm, based on time series modeling and multivariate statistical techniques, decomposes the real-time data from equipment sensors into two components and produces a novel double T^2 control chart for SPC. Examples using RTSPC, the software utility implementing this new fault detection algorithm, were shown for fault detection on data collected from various plasma etchers. In addition, an algorithm combining principal component regression and ARIMA modeling has been investigated to extend the algorithm to include multiple recipes.

References for Chapter 4

[4.1] H. Guo, *Real Time Statistical Process Control for Plasma Etching*, Master's Thesis, University of California, Berkeley, Memorandum No. UCB/ERL M91/61, July 1991.

[4.2] C. J. Spanos, H. Guo, A. Miller and J. Levine-Parrill, "Real-Time Statistical Process Control Using Tool Data," *IEEE Trans. Semiconductor Manufacturing*, vol. 5, no. 4, Nov. 1992, pp. 308-318.

[4.3] D. C. Montgomery, *Introduction to Statistical Quality Control*, 2nd ed., John Wiley & Sons, 1991.

[4.4] Western Electric *Statistical Quality Control Handbook*, Western Electric Corp., Indiana, 1956.

[4.5] A. Pankratz, *Forecasting with Univariate Box-Jenkins Models*, John Wiley & Sons, 1983.

[4.6] P. J. Brockwell, R. A. Davis, *Time Series: Theory and Methods*, 2nd ed., Springer-Verlag, 1991.

[4.7] R. H. Shumway, *Applied Statistical Time Series Analysis*, Prentice Hall, 1988.

[4.8] D. R. Brillinger, *Time Series: Data Analysis and Theory*, Holt, Rinehart, and Winston, 1975.

[4.9] C. J. Spanos, S. Leang, S. F. Lee, "A Control and Diagnosis Scheme for Semiconductor Manufacturing," *American Control Conference*, vol. 3, June 1993, pp. 3008-3012

[4.10] H. C. Liu, "Automatic Time-Series Model Generation for Real-Time Statistical Process Control," UCB/ERL M93/45, June 8, 1993.

[4.11] S. F. Lee, E. D. Boskin, H. C. Liu, E. Wen, C. J. Spanos, "RTSPC: A Software Utility for Real-Time SPC and Tool Data Analysis," to appear in *IEEE Trans. Semiconductor Manufacturing*.

[4.12] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, *Time Series Analysis: Forecasting and Control*, 3rd ed., Prentice Hall, 1994.

[4.13] W. Vandaele, *Applied Time Series and Box-Jenkins Models*, Academic Press, 1983.

[4.14] B. L. Bowerman, R. T. O'Connell, *Forecasting and Time Series: An Applied*

Approach, 3rd ed., Duxbury Press, 1993.

[4.15] *S-PLUS User's Manual*, version 3.0, Statistical Sciences, Inc., Seattle, WA, Sept. 1991.

Chapter 5

Fault Diagnosis

5.1 Introduction

There are several benefits of having a real-time diagnosis system. First, by detecting and troubleshooting faults while the wafer is being processed, improvements in the capability and uptime of critical process equipment are possible. Second, a diagnostic system can provide early indication of impending malfunctions, or prognosis, so that potential problems can be corrected before a catastrophic failure occurs. An advantage of using real-time tool data is that it can be collected automatically and inexpensively, and can be used either independently or with other information, such as wafer measurements.

Once the Fault Detection Module described in the previous chapter has detected an equipment fault, the residuals from the long-term component are siphoned to the Fault Diagnosis Module. These residuals form a signature that can be traced back to a specific equipment fault or group of faults. The types of faults identified include changes in the

input settings such as pressure, RF power, and flow changes. Just as the fault detection module requires training to recognize in-control, or baseline, behavior, the fault diagnosis module requires training to recognize the signature of "faulty" signals. Thus, the module is first trained by injecting known faults into the equipment. During production, when a new fault is detected the fault diagnosis module will recognize the signature of the long term component residuals and diagnose the root cause of the fault.

While the real-time residuals of the long term component contain relevant information, diagnosis of the faults is not straightforward. For example, Figure 5.1 shows a two-dimensional plot of the coil position residuals versus the impedance residuals. The two fault clusters shown can not be easily distinguished by projecting the data on either of the axes. Thus, two methods for fault diagnosis were developed. The first uses discriminant analysis techniques. While this method has shown promise, it is not scale invariant and potentially requires many training runs. The second method, which we call staged clustering and neural network analysis, overcomes this problem at the expense of more complex training. Before describing the different fault diagnosis methods, however, the method used to measure of accuracy of each diagnosis system is described.

5.2 Probabilities of Misdiagnosis

Once the module has been trained, it is important to know the accuracy of the diagnosis system, or the probability of misdiagnosis. In general, the probability of misdiagnosis is defined as the probability of incorrectly allocating an individual point from fault population $\prod_i$ to population $\prod_j$ and is denoted as p_{ij}. The estimate of p_{ij} is [5.1]

$$\hat{p}_{ij} = \frac{n_{ij}}{n} \tag{5.1}$$

where n_{ij} is the number of individuals from fault population $\prod_j$ which were allocated to one of the other fault populations $\prod_i$ ($i \neq j$), and n is the total number of points. In other

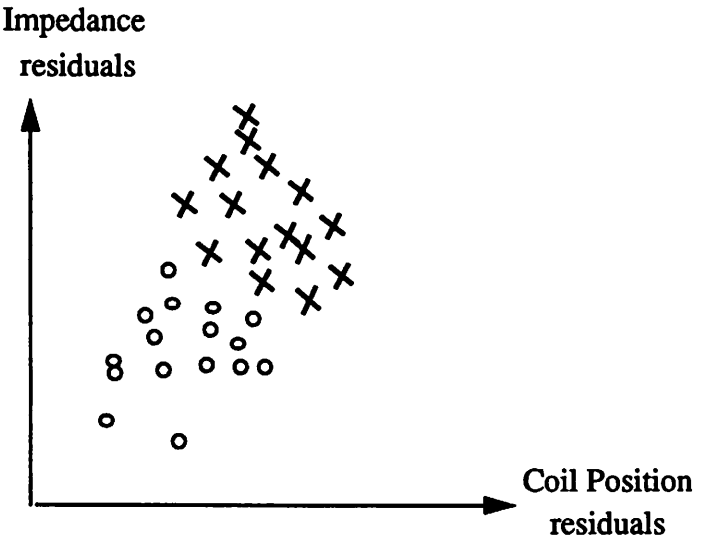


Figure 5.1 Plot of Coil Position Residuals vs. Impedance Residuals from the long term components. Two distinct fault types are shown.

words, the estimate of the probability of misdiagnosis is given by the ratio of the number of misallocated faults to the total number of faults n. Three methods to test the validity of the discrimination, resubstitution, cross-validation, and an independent test set, are described in this section.

5.2.1 Resubstitution

The most straight-forward method, called resubstitution, is to test the system with the same data set used to train it [5.2]. An estimate of the probability of misdiagnosis is simply the ratio of incorrectly diagnosed points to the total number of points. Because the same data set used to train the system is also used to evaluate the diagnostic capability, this method only describes how well the original data were trained and gives no information about the actual accuracy of diagnosis of other data. Therefore, the resubstitution method tends to result in optimistic misdiagnosis probabilities [5.3][5.4].

5.2.2 Cross-Validation

Cross-validation, also known as the U-method or jack-knifing, is more objective than resubstitution [5.4]. In this method, one point is taken out of the training analysis, and then used to test the system. This occurs for each set of observations, so that for a total of n observation sets, n different discriminant rules are determined using a subset of (n -1) observation sets. This method results in more accurate estimates of p_{ij} than the resubstitution method for multinormal populations with the same covariance matrices.

5.2.3 Independent Test Set

The most reliable method to determine the probability of misdiagnosis is to test the system on an independent data set not used to generate the discrimination rule. The only criterion is that the test set should be representative of possible faults that the equipment may encounter. Although it requires more experiments, this method is the most accurate of the three, and thus was chosen for this work.

5.3 Diagnosis Based on Discriminant Analysis

5.3.1 Theory

The analysis in this section is based on a diagnostic algorithm which uses discriminant analysis techniques to analyze the long-term component residual data. In general, discriminant analysis techniques classify a set of measurements into one or more known populations. In this case, the long-term component residuals comprise the measurement sets and the populations are specific equipment faults. There are several methods to perform discriminant analysis, depending on the distributions of the residuals and available information.

The simplest case to analyze is one in which the exact probability density functions (p.d.f.s) are known [5.1]. Although this is rarely seen in experimental work, the distribu-

tions can be estimated fairly accurately for large samples. The training data sets for equipment faults, however, are rather small and number fewer than 20 points. Furthermore, for this application the cost of producing enough samples to obtain distribution estimates is prohibitive.

Another case occurs when the overall form of the distributions are known, but certain parameters of the distributions must be estimated. Two methods to perform discriminant analysis for this case are maximum likelihood and likelihood ratio methods. If the probabilities of each fault population are known *a priori*, Bayesian methods can also be utilized. For the data set used in this application, however, prior probabilities are not known. It is conceivable that if enough runs are performed on the machine, and the faults are tracked and categorized, reasonable *a priori* probabilities for certain faults can be obtained. This requires a large number of runs which becomes extremely expensive in a semiconductor manufacturing environment.

Tests for normality such as the kurtosis or skewness tests show that the data can not be assumed to be normally distributed. More importantly, the nature of the problem at hand does not fit well into the maximum likelihood, likelihood ratio, or Bayesian methods. In this application, the actual real-time residual signatures are mapped directly to the fault. Since this signature is fixed and does not change from run to run, none of the above methods is the best approach.

5.3.2 Training via Fisher's Linear Discriminant Method

Instead, the method of choice for this thesis is Fisher's linear discriminant method, which assumes nothing about the distributions and instead finds a reasonable method to discriminate the groups. Fisher's linear discriminant analysis compares the variances among populations to the variance within a certain population. For discrimination to occur, the ratio of the variances should be significant. More formally, we determine the

vector $\mathbf{a_1}$ which maximizes the ratio of the between-groups sum of squares (**B**) to the within-groups sum of squares (**W**) of data matrix $\mathbf{X}= (\mathbf{x_1}\ \mathbf{x_2}\ \mathbf{x_3}...)^T$ [5.1]:

$$max\frac{\mathbf{a_1}^T\mathbf{B}\mathbf{a_1}}{\mathbf{a_1}^T\mathbf{W}\mathbf{a_1}} \tag{5.2}$$

$$\mathbf{B} = \sum n_i (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})\ (\bar{\mathbf{x}}_i - \bar{\mathbf{x}})^T \tag{5.3}$$

$$\mathbf{W} = \sum n_i \mathbf{S}_i \tag{5.4}$$

where $\mathbf{x_j}$ (n_i x p) represents n_j observations from fault population $\prod_j$. $\mathbf{S}_i$ is the sample covariance matrix, $\bar{\mathbf{x}}_i$ is the sample mean, and n_i is the number of samples in each fault group, so that $n = \sum_i n_i$.

The vector $\mathbf{a_1}$ which maximizes the above ratio is the eigenvector of $\mathbf{W}^{-1}\mathbf{B}$ corresponding to the largest eigenvalue. The set of linear discriminant functions is $\mathbf{Xa_1}$, $\mathbf{Xa_2}$, $\mathbf{Xa_3}$,..., $\mathbf{Xa_p}$where $\mathbf{a_2}$ is the eigenvector corresponding to the second largest eigenvalue of $\mathbf{W}^{-1}\mathbf{B}$, $\mathbf{a_3}$ is the eigenvector corresponding to the third largest eigenvalue, and so on. Thus, $\mathbf{Xa_1}$ is the linear combination of the signals that will maximally discriminate among the faults in one dimension.

5.3.3 Diagnosis

Once the linear discriminant functions are found, classification of the populations can be performed using the Euclidean distance scheme. In one dimension, an observation **x** is allocated to one of the populations based on its "discriminant score" $\mathbf{a_1x}$ [5.1]. The sample means $\bar{\mathbf{x}}_i$ have scores $\mathbf{a_1}\bar{\mathbf{x}}_i$. Observation **x** is allocated to fault population $\prod_j$ if

$$\left|\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\bar{\mathbf{x}}_j\right| < \left|\mathbf{a}^T\mathbf{x} - \mathbf{a}^T\bar{\mathbf{x}}_i\right| \quad \text{for all } i \neq j. \tag{5.5}$$

This can easily be extended to several dimensions. Essentially, the boundary among faults is determined by a hyperplane equidistant from the geometric mean of each fault in the space of the linear discriminant functions. Figure 5.2(a) shows two fault clusters in the

space of two long-term component residuals. Axis $\mathbf{Xa_1}$ is the first linear discriminant function which maximizes the ratio of the between groups sum of squares to the within groups sum of squares of the two faults. The geometric mean of each fault is then calculated, and an equidistant line determines the boundary between the faults. The new point can be easily diagnosed. Discrimination of three faults using two linear discriminant functions is shown in Figure 5.2(b). The geometric means are shown, along with the classification boundaries. Once again, the new faulty point can be diagnosed. For some cases, more dimensions are necessary for maximal discrimination.

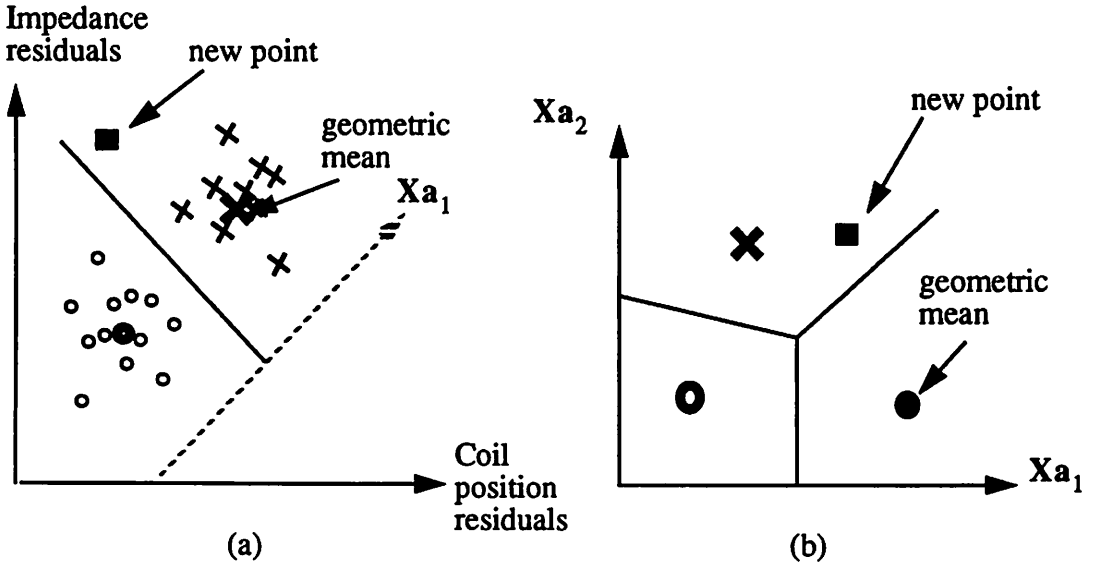


Figure 5.2 Examples of fault diagnosis. (a) The two fault clusters in the space of the long-term component residuals. Axis $\mathbf{Xa_1}$ is the first linear discriminant function. The geometric mean of each fault is then calculated, and an equidistant line determines the boundary between the faults. In this way, the new point can be easily diagnosed to the proper fault. (b) The geometric means of three faults in the space of the first two linear discriminant functions. The new data point can also be diagnosed.

5.3.4 Examples Using Discriminant Analysis

The example in Figure 5.3 shows discrimination among six equipment faults (labelled 1-6) in a Lam Rainbow 4600 metal plasma etcher. The faults are individual runs in a fractional factorial design, where RF forward power, chamber pressure, Cl_2 gas flow, and the He backflow behind the wafer were varied ±5 %. During the etcher operation, seven tool signals were monitored: RF tune vane position, RF coil, RF phase, RF impedance, DC bias, peak voltage, and an optical endpoint emission signal. These signals were monitored using the LamStation software at a rate of approximately one sample per second. The diagram shows the six regions used to classify the six faults in the space of the first two linear discriminant functions.

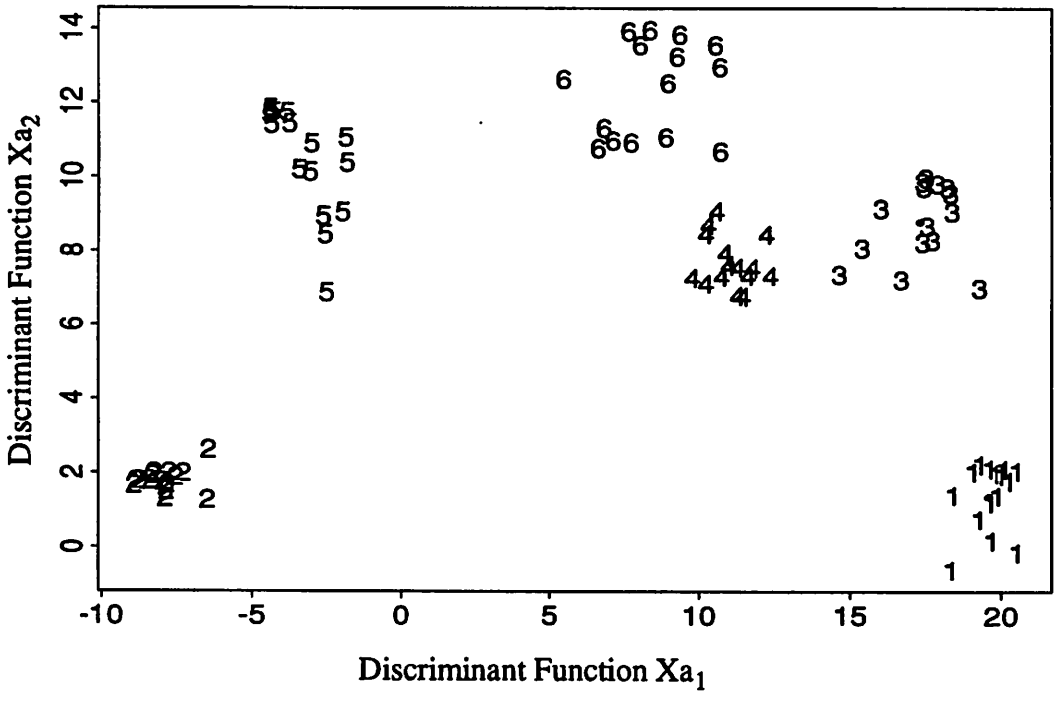


Figure 5.3 Training of Six Faults (labelled 1-6) using Discriminant Analysis. The projection in the space of the first two linear discriminant functions is shown. This example was conducted on a Lam Rainbow 4600 metal plasma etcher.

After the discriminant rule was determined during the training stage, it was tested with additional wafers run at the same operating conditions as faults numbered 5 and 6. The projection in the space of the first two linear discriminant functions is shown in Figure 5.3. The geometric means of the training runs are also shown in the figure. The diagnosis algorithm was performed in three dimensions, so that the faults labelled 6 and 4 could be distinguished. In three dimensions, the geometric mean of fault number 6 falls above the page, while that of fault number 4 is below the page. The misdiagnosis rate in three dimensions is 10% using this independent test set. (The resubstitution method estimated an optimistic 0% misdiagnosis probability.)

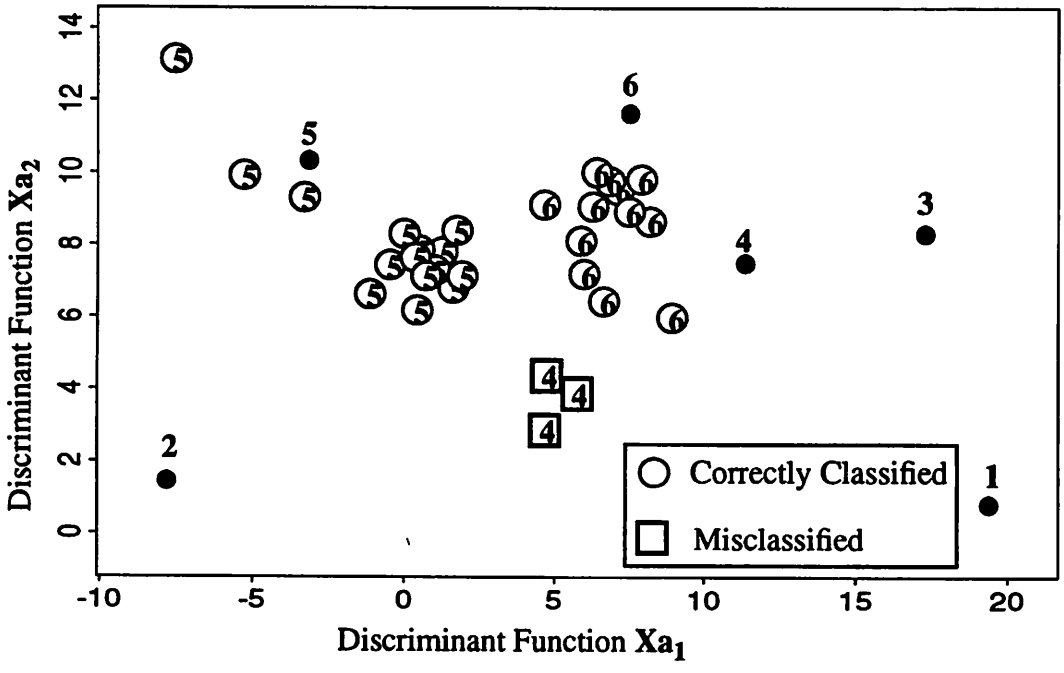


Figure 5.4 Diagnosis of two types of faulty runs, corresponding to faults 5 and 6. The projection in the space of the first two linear discriminant functions is shown. Note that the means from the training runs are also shown. This example was conducted on a Lam Rainbow 4600 metal plasma etcher.

The second example shows diagnosis of single faults. The system was trained with the data from the Training Phase II Experiment described in section 3.2.2. The seven single faults were ±20 % changes in chamber pressure, electrode gap spacing, and gas flow ratio, and a 20% increase in RF power. Later, a wafer processed with a 20% decrease in gas flow ratio was correctly diagnosed, as shown in Figure 5.3. In three dimensions, thirteen runs were diagnosed properly, and two were diagnosed improperly as fault 4, corresponding to a misdiagnosis rate of 13.3%.

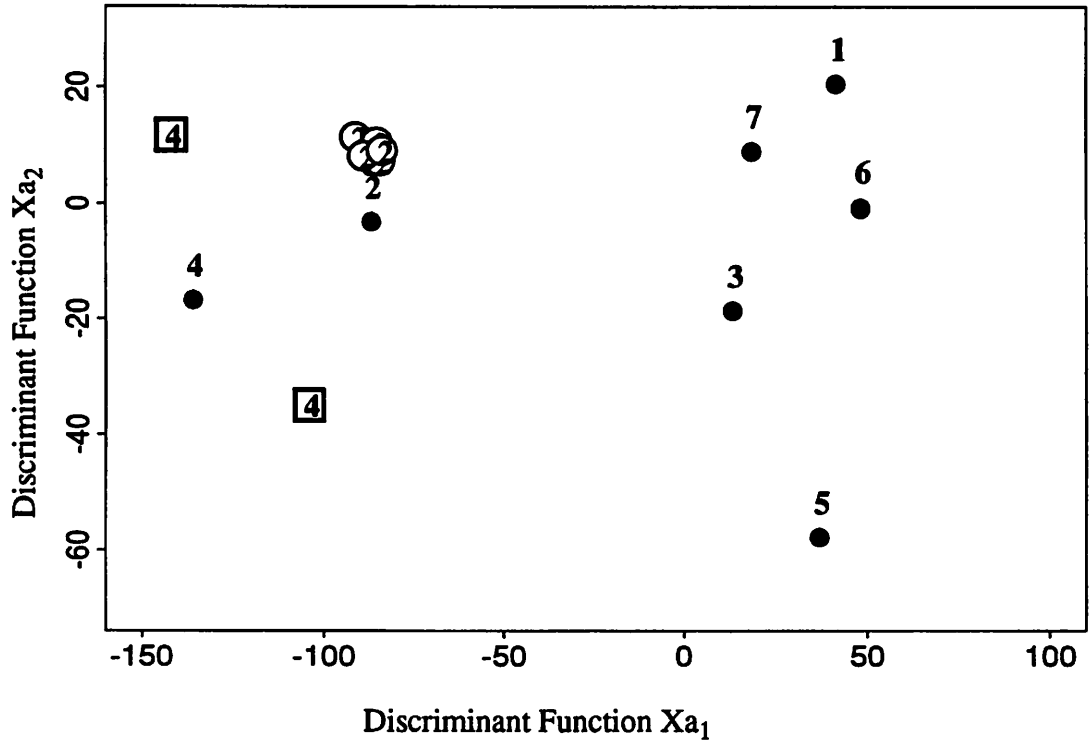


Figure 5.5 Diagnosis of single fault (increase in the ratio of the gas flows by 20%), corresponding to fault 2. The projection in the space of the first two linear discriminant functions is shown. Note that the means from the training runs are also shown. This example was conducted on a Lam Rainbow 4400 polysilicon plasma etcher.

Thus by employing time-series models and discriminant analysis techniques, real-time sensor data can be used effectively to detect and classify equipment faults. The main

advantages of using discriminant analysis techniques is that training the system is easy, since it simply requires a set of long-term component residuals and known faults. The user does not need to be an expert on the system, as the algorithm is completely data-driven. The computations are simple matrix algebra functions, which are fast on modern computers. Therefore, it is easy to maintain a library of faults. For example, if a new fault is seen by the module and can not be properly classified, the real-time residual data can be stored until the technician has diagnosed the fault. Once the fault has been correctly diagnosed, that information can be fed back into the training data set. Then a new discriminant rule can be calculated to include the new fault.

The major disadvantage with the discriminant analysis technique is that it is not scale invariant. For example, if the module has been trained to recognize a change in RF power of 20%, it will not recognize a change in RF power of 10%. Therefore, the system must be trained to recognize the signature of each fault, which potentially requires many training runs. This problem is addressed by the second diagnostic method discussed in the following section.

5.4 Diagnosis Based on Staged Clustering and Neural Network Analysis

The second method developed for equipment diagnosis, which uses staged clustering and neural network analysis, diagnoses various levels of equipment faults while requiring few training runs. Unlike the previous method using Fisher's discriminant analysis function which was easily trained, this method requires slightly more complex and interactive training. The general idea is that clustering techniques exploit the trends in the long-term component residuals to diagnose certain faults, while the neural networks extract the finer details in the data to diagnose other faults. For the Lam Rainbow 4400 plasma etcher, the clustering techniques have trouble separating changes in chamber pressure from changes

in individual gas flows, so neural networks are employed to separate these two particular faults. A brief discussion of cluster analysis and neural networks follows.

5.4.1 Clustering Methods

Clustering algorithms group similar objects, and are used in this section to perform the diagnosis and prognosis of equipment faults. These algorithms are heuristic in nature, and are purely data-driven. Furthermore, there are few standard measures of clustering validity. Despite the heuristic nature of cluster analysis, we found it to be quite promising in diagnosing and prognosing equipment faults based on the data set obtained from the Training Phase II, Verification, and Diagnosis Experiments, described in Chapter 3.

First, a vector of measurements which characterize the objects to be clustered is determined. In this case, the vector consists of the real-time signals collected from the equipment, and is referred to in this chapter as the "real-time signals vector." Next, a similarity metric on which to base similarity or dissimilarity among data points is determined. These include several well-known distance metrics such as the Euclidean, Manhattan (absolute), and Mahalanobis distances. In this module, the Euclidean distance method resulted in the most effective clustering, where the distance between data vectors i and j with p parameters is defined as:

$$d_{ij} = \sqrt{\sum_{k=1}^{p} (x_{ik} - x_{jk})^2} \tag{5.6}$$

where x_{ik} is the value of the kth term in the ith vector [5.5].

Two commonly used clustering techniques are the hierarchical and optimization methods. In hierarchical clustering, the data is separated in stages. Once a particular point has been separated into a cluster, it can not be reallocated. On the other hand, optimization techniques allow the data to be reallocated through an iteration. Many of these techniques, however, require that the number of distinct clusters in the data be known, and that the

clusters are spatially homogeneous. Because it is not known exactly how many faults are in the data set, hierarchical clustering was chosen for this thesis. Within hierarchical clustering techniques, the two main methods used to cluster the data are the agglomerative and the divisive methods. Agglomerative methods fuse individuals or groups of data which are the closest by some measure, while divisive methods split groups successively into smaller clusters. Due to the nature of the data in the diagnostic module, agglomerative methods were investigated.

Many agglomerative clustering methods exist, of which two are described here. For a more thorough discussion of the other methods, [5.5], [5.6], and [5.7] are excellent sources. The nearest-neighbor, or single-link, method connects points or groups based on the distance between their nearest neighbors. This method tends to result in groups with a small number of members, "chained" together by single links. Because it results in many small groups linked together rather than a few larger groups, this method is unsatisfactory for diagnosis, where data points need to be grouped distinctly into fault clusters. The furthest-neighbor, or complete linkage, method has the opposite algorithm, in which clusters are formed based on the distance of the farthest neighbors of each group. Because this method results in larger groups of clusters that are easily separated, it is used in the diagnostic module.

5.4.2 Neural Networks

Neural network models are empirically-based models which train a combination of "neurons," or nodes, to learn and model relationships between a set of inputs and outputs. The connections among the nodes are weighted. Each node receives a net input computed from the sum of the weighted outputs of the nodes preceding it, "squashed" by an activation function. A common activation function used for each node is a logistic function of

the form $f(x) = \frac{1}{1+e^{-x}}$. The output of the node can also be transformed by a function, which is usually taken to be the identity function.

There are three types of nodes; those whose inputs are the inputs of the problem are called the input nodes, and make up the *input layer*; those whose outputs are the output of the problem form the *output layer;* the nodes connecting the input and output nodes form the *hidden layer.* These three layers are depicted in Figure 5.6, which shows three nodes in the input layer, and two nodes in the hidden and output layers. Also shown are connections between the input nodes and the hidden nodes, and between the hidden nodes and the output nodes.

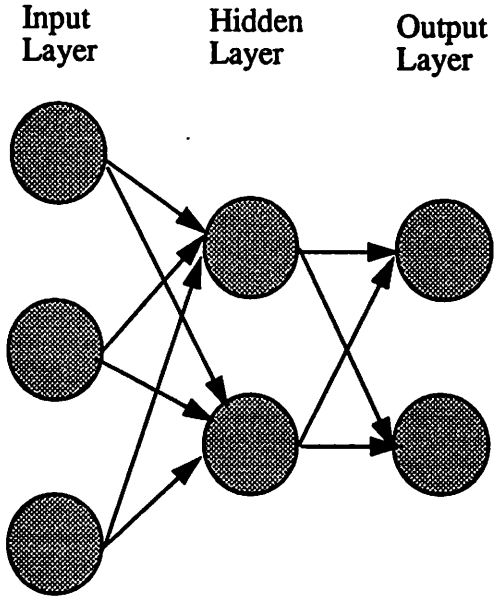


Figure 5.6 Small neural network with three layers of units. The figure shows three input nodes, one hidden layer with two nodes, and two output nodes. The connectivity used is between the input units and hidden layer, and between the hidden layer and output units.

The neural network algorithm selected for this analysis is the feed-forward, error back-ward propagation (FFEBP) method, which has shown to be effective in modelling noisy

input and output data [5.8][5.9]. In this algorithm, the inputs are fed forward through the layers of the network until reaching the output layer. The result at the output layer of node *j* is compared with the desired, or teaching, output. The difference, called the error, is used with the output of node *i* to calculate the new weighting of the connection between node *i* and node *j*. These errors are then used to calculate the weight changes for the connection between the input and hidden units. Because the weight corrections depend upon the corrections previously computed from the neighboring layer, the error in effect is propagated backward through the network [5.10]. In the FFEBP method, the gradient search method is used to minimize the sum of the squared errors [5.11].

For this application, we have selected to use a FFEBP neural network with one input layer, one hidden layer, and one output layer. An experimental analysis led to a network with 13 nodes in the input layer, 8 nodes in the hidden layer, and two nodes in the output layer. The connectivity chosen is between the input units and hidden layer, and between the hidden layer and output layer. The Stuttgart Neural Network Simulator (SNNS) was used to simulate and train the neural networks [5.10]. The network learns the relationship between the input and output patterns as it undergoes learning iterations. To determine when to stop training, a separate testing data set was used. Training stopped when the testing set achieved its lowest error. This is a usual practice to eliminate over-training, which results in decreased generalization capability of the network model.

5.4.3 Pre-filtering of the Long-term Component Residuals

Before performing training and diagnosis, the residuals obtained from the long-term time series models (section 4.3) are pre-filtered to determine which data are significantly different from the baseline process. This test of statistical significance is performed by using the student-t test, with the significance level of 0.01:

$$\frac{x - \mu_{cent}}{\sigma_{cent}} \sim t_{\alpha v}$$

where μ_{cent} is the mean of the centerpoint data, σ_{cent} is the standard deviation of the centerpoint data, α is the significant level for the test, and ν is the degrees of freedom. The residuals which are not statistically significantly different from zero are replaced with a value of zero. The average real-time readings per wafer run are filtered for the clustering stages. The resulting filtered real-time readings make up the real-time signal vector used to calculate the distance between groups of faults. Fifteen points per wafer are used to train the neural network.

5.4.4 Training

To train the system, a set of heuristics is developed specifically for each type of machine. The idea is to determine the distance of the real-time signal vector as calculated in Equation (5.6) among two or more runs of a particular type of fault (for example, Fault A), and the other faults. The midpoint between the largest distance among runs of Fault A and the smallest distance between Fault A and the other faults is considered to be the cut-off value. During diagnosis, new runs with distances less than the cut-off value are assigned to Fault A. These distance measures and corresponding cut-off values are calculated for each training fault, requiring at least two training runs per fault type. When limited training data is available, for example if no replicated points are conducted in a central composite design with star points, the same class of faults are grouped together regardless of magnitude. Then the signs of one of the runs are flipped so that both an increase and decrease in the particular parameter have the same signal characteristics. For example, if the training data contains only one run with an increase in power and one run with a decrease in power, the signs of the latter run are flipped so that the data from both now correspond to an increase in power. The diagnosis, then, will simply indicate which parameter has changed, and will not indicate whether it was an increase or decrease from nominal.

To group the faults, the filtered data is first converted to +1, -1, or 0 if the value of the residual is positive, negative, or zero, respectively. Upon clustering, the data forms two groups, separating the faults between positive and negative changes from the nominal value. The signs of those faults with negative changes are flipped, so that all the faults, regardless of magnitude, are in one group. This simplifies the analysis, and can be done when the real-time signals are either monotonically increasing or decreasing around the centerpoint data. Otherwise, two sets of analysis can be performed so that increasing and decreasing faults can be diagnosed separately.

In the case of the Lam Rainbow 4400 plasma etcher, the trends in the real-time signals are used to distinguish among three groups: runs with no faults (center points), those with RF power problems, and the rest. Table 5.1 shows the trends for a subset of the long-term component residuals from the RPM-1 signals for various single equipment faults on a Lam Rainbow 4400 plasma etcher. Each of the listed faults are increasing from nominal levels; for example, the faults in the table include an increase in RF power, chamber pressure, ratio of the two gases, or electrode gap spacing. The data was collected during the Training (Phase II) and Diagnosis Experiments, as described in sections 3.2.2 and 3.2.3. The table shows that an increase in RF power is captured by the signals as an increase in delivered RF power, RMS voltage, and RMS current, and a decrease in the phase angle and DC bias, while an increase in chamber pressure leads to an increase in RF power, phase angle, and DC bias, and a decrease in RMS voltage and RMS current. It is interesting to note that although it is expected that an increase in RF power leads to an increase in the measured RF power, an increase in chamber pressure or gas flow also result in an increase in the measured RF power, illustrating why multiple signals are necessary for fault diagnosis.

Table 5.1 Trends of Long-Term Component Residuals for Various Equipment Faults on a Lam Rainbow 4400 plasma etcher

Fault (increase)	Trends of Long-Term Component Residuals				
	RF Power	Voltage	Current	Angle	DC Bias
RF Power	↗	↗	↗	↘	↘
Pressure	↗	↘	↘	↗	↗
Cl_2 Flow	↗	↘	↘	↗	↗
Gap	↘	↗	↗	↘	↘
Total Flow	↗	↘	↘	↗	↗

Table 5.1 also shows that trends are not enough to distinguish among all the faults. An increase in either the chamber pressure or the gas flow, and a decrease in the electrode gap spacing have the same trends. This is also true when the data collected via LamStation is included. Thus, the magnitude of the signals must be used to diagnose these faults, including faults with the electrode gap spacing and the total gas flow. Other faults can not be diagnosed by staged clustering, since the trends and magnitudes of the signals are so similar. For example, decreases in chamber pressure are confused with decreases in the Cl_2 gas flow. For these cases, neural networks are used to model the subtle differences between the signal sets, as explained next.

Standard FFBEPNN was used on 15 readings per wafer for training and during diagnosis. First, the pre-filtered real-time residuals are scaled so that the ranges of both the input and output are between 0 and 1. The training output is set at 0 for decreases from nominal, 1 for increases, and 0.5 for normal behavior.

5.4.5 Diagnosis/Prognosis

Both the data used for training and the heuristic limits derived in the training stage are needed for diagnosis. The training data are used to represent the various faults, while the

heuristics determine the fault group to which the new point belongs. To begin the diagnosis, the trends of the data are used. The faults are determined in consecutive stages as outlined in Figure 5.7. The first two stages classify faults with distinct trends in the filtered real-time data. First, the algorithm checks to see if the data is from a normal run. Since after pre-filtering most of the parameters from centerpoint runs will be set to zero in the real-time signal vector, they can easily be distinguished from the faulty runs. If the algorithm determines that the new data is not a centerpoint run, it then checks for a fault in the RF power by examining the trends in the filtered real-time data. If the RF power seems normal, the training data corresponding to faults in the RF power are eliminated, and the remaining data is sent to the next stage. The algorithm then looks for other faults, one at a time. Because the other faults can not be distinguished solely by trends in the real-time signals, the magnitudes of the filtered real-time data are clustered for the remainder of the analysis. The first is electrode gap spacing, which uses the DC Bias readings. Voltage, current, impedance, phase angle, DC Bias, endpoint, and phase error readings are all required to separate changes in total gas flow from changes in chamber pressure and Cl_2 gas flows. Once again, after each stage, the training data corresponding to each of the tested faults are removed from the data set.

At this point, the data is sent to the neural network model, which has been trained to recognize the remaining faults. For the Lam Rainbow 4400, these include changes in chamber pressure and Cl_2 gas flows. The neural network assigns one fault to each point. Because 15 points are associated with each wafer, the final fault assigned to the wafer is the fault which has been assigned eight or more times to that wafer.

5.4.6 Example Using Staged Clustering and Neural Network Analysis

In this example, two sets of training experiments were conducted. The first is the Training Phase II Experiment (section 3.2.2), in which wafers with single faults of approx-

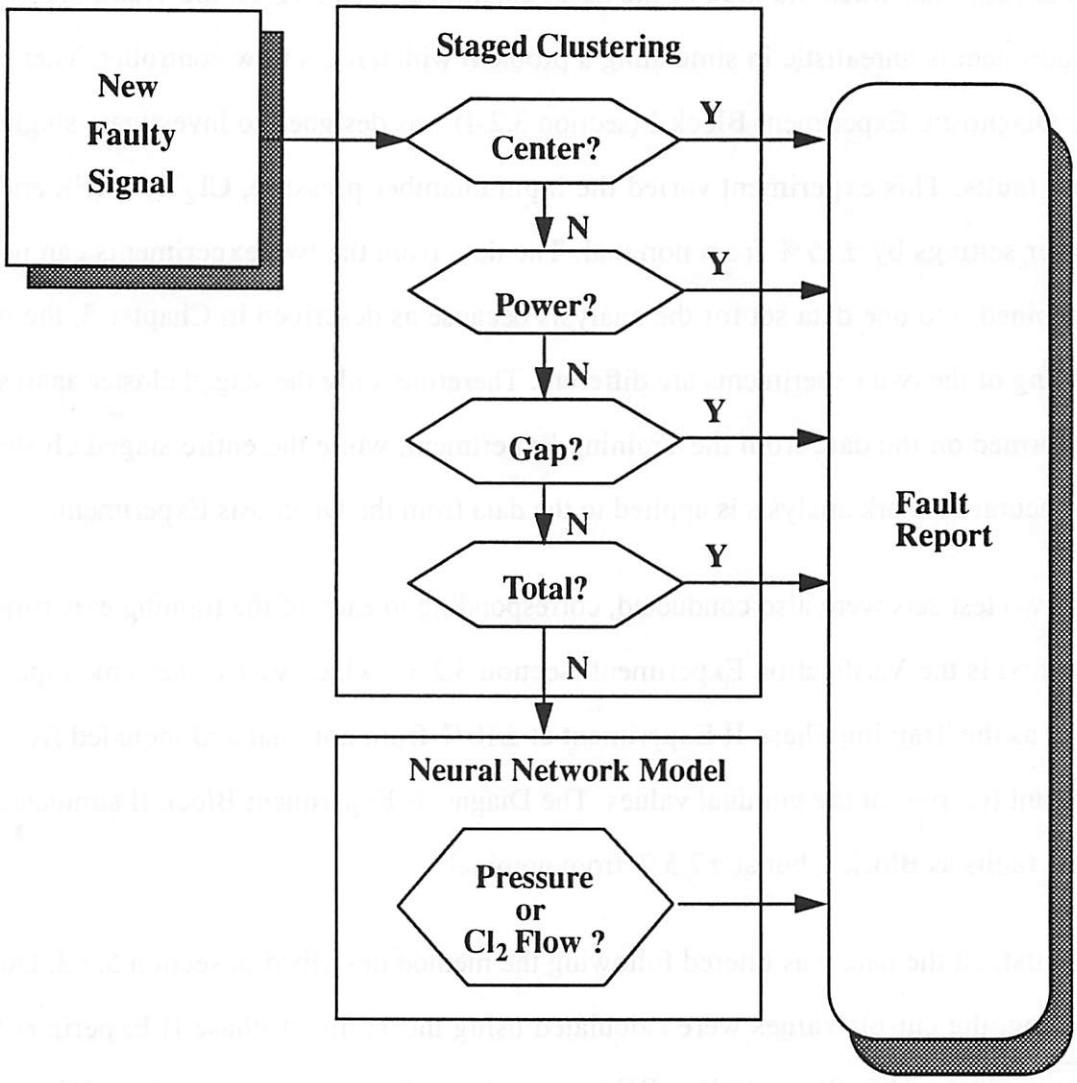


Figure 5.7 Diagnosis Using the Staged Clustering and Neural Network Technique for the Lam Rainbow 4400. Clustering methods are used to diagnose problems with RF power, electrode gap spacing, and total flow, while a straight forward neural network is used to distinguish between chamber pressure and Cl_2 gas flow changes.

imately ±20 % from nominal were etched. The input settings which were varied are the chamber pressure (P), the ratio of the gases (R), the RF power (W), the electrode gap spacing (G), and the total flow of the gases (T). Because it is unlikely that the total flow will

remain constant when the ratio of the gases changes and vice versa, the Training Phase II Experiment is unrealistic in simulating a problem with a mass flow controller. Therefore, the Diagnostic Experiment Block I (section 3.2.4) was designed to investigate single gas flow faults. This experiment varied the input chamber pressure, Cl_2 flow (F), and RF power settings by ±15 % from nominal. The data from the two experiments can not be combined into one data set for the analysis because as described in Chapter 3, the mask loading of the two experiments are different. Therefore, only the staged cluster analysis is performed on the data from the Training Experiment, while the entire staged clustering and neural network analysis is applied to the data from the Diagnosis Experiment.

Two test sets were also conducted, corresponding to each of the training experiments. The first is the Verification Experiment (section 3.2.3), which varied the same input settings as the Training Phase II Experiment at ±10 % from nominal and included five centerpoint (C) runs at the nominal values. The Diagnosis Experiment Block II simulated the same faults as Block I, but at ±7.5 % from nominal.

First, all the data was filtered following the method described in section 5.4.3. During training, the cut-off values were calculated using the Training Phase II Experiment for centerpoints and faults including RF power, electrode gap spacing, and total flow. The neural network was trained to recognize the chamber pressure and gas flow changes from the Diagnostic Experiment Block I. We used a feed-forward error back propagation neural network (FFEBPNN) with 13 input nodes, one hidden layer with eight nodes, and two output nodes, one for changes in chamber pressure, the other for changes in gas flows.

The data from the Verification Experiment were used to test the clustering stages of the algorithm. The results are shown in Table 5.3. The first column of each group indicates the type and magnitude of the fault injected into the machine, while the second column shows whether the fault was correctly (✓) or incorrectly (X) diagnosed. Because the faults were

initially grouped, the diagnosis simply states which input parameter has changed, and does not give indication if the faulty parameter has shifted up or down.

The results show that all wafers run at the centerpoint values were correctly diagnosed as having no faults. In addition, changes in both RF power and electrode gap spacing were correctly diagnosed. Furthermore, a decrease in total flow was diagnosed as such. An increase in the total flow, however, was misdiagnosed as having no fault. Upon examination of the signals, it was found that none of the signals differed from the centerpoint values. In fact, the final etch rates, selectivities, and uniformities of that particular wafer did not indicate a faulty condition, so the staged clustering algorithm was indeed correct in diagnosing the run as a centerpoint.

Table 5.2 Diagnosis After Clustering Stages: Verification Experiment

Fault	Diag?
C	✓
C	✓
W (+ 15%)	✓
G (+ 15%)	✓
T (+ 15%)	×

Fault	Diag?
C	✓
C	✓
W (- 15%)	✓
G (- 15%)	✓
T (- 15%)	✓

The neural network stage, which classifies the run as having a problem with either the chamber pressure or the gas flow, was tested with the data from Block II of the Diagnosis Experiment. Because the experiment also contained centerpoints and runs with RF power changes, the data was first pre-filtered and sent through the clustering stages. The same algorithm used for the previous experiment was applied successfully to this new data set to isolate those runs at centerpoint conditions and those with changes in the RF power. The fact that the same algorithm held for both experiments shows a strength of the

method, since it can be applied to two sets of data with different wafer loading due to patterning differences (section 3.2).

Once both centerpoint and RF power data were removed from the data set, the data were sent to the neural network. The results of the clustering and neural network stages are listed in Table 5.3. Both centerpoints were diagnosed properly using cluster analysis, as were the runs with RF power changes. The neural network diagnosed all changes in Cl_2 flows properly, as well as three of four chamber pressure changes.

Table 5.3 Diagnosis After Staged Clustering and Neural Networks: Diagnosis Experiment

Fault	Diag?	Fault	Diag?
C	✓	C	✓
W (+ 7.5%)	✓	W (- 7.5%)	✓
W (+ 7.5%)	✓	W (- 7.5%)	✓
P (+ 7.5%)	✓	P (- 7.5%)	✓
P (+ 7.5%)	×	P (- 7.5%)	✓
F (+ 7.5%)	✓	F (- 7.5%)	✓
F (+ 7.5%)	✓	F (- 7.5%)	✓

On the whole, the results of the staged clustering and neural network method are very promising. Because the method is scale invariant, it requires fewer runs for training than using discriminant analysis. The experiment used for training is a simple star design, which is generally conducted by the fabs during the qualification of the machines to choose a proper operating point, so this method does not require the fab to conduct additional runs strictly for diagnosis purposes. Although it requires longer training, the staged clustering and neural network algorithm successfully diagnosed faults which were at different levels from the training faults.

5.5 Fault Diagnosis Module Summary

The difficult and lengthy process of trouble-shooting equipment faults makes diagnostic capability an important addition to SPC. Two methods for equipment diagnosis were developed and demonstrated in this chapter. The long-term component residual from the fault detection module are used in both diagnosis methods. The first employs Fisher's discriminant analysis techniques to separate faults. The method has been demonstrated on both single and multiple faults. This method enjoys certain advantages, such as fast and simple training. The disadvantage of the method is that it is not scale invariant, so it requires many training runs. The second method, using staged clustering and neural network analysis, is scale invariant. Examples of single faults at different levels from the training faults were shown. Coupled with the benefits of real-time SPC, diagnosis will greatly reduce the cost of ownership of manufacturing equipment.

References for Chapter 5

[5.1] K. V. Mardia, J. T. Kent, J. M. Bibby, *Multivariate Analysis*, Academic Press, 1979.

[5.2] C. A. B. Smith, "Some Examples of Discrimination," *Ann. Eugen.*, vol. 18, 1947, pp. 272.

[5.3] M. Hills, "Allocation Rules and Their Error Rates," *J. Roy. Stat. Soc.*, B28, p. 1.

[5.4] P. A. Lachenbruch, M. R. Mickey, "Estimation of Error Rates in Discriminant Analysis," *Technometrics*, vol. 10, no. 1, Feb. 1968, pp. 1-11.

[5.5] B. Everitt, *Cluster Analysis*, second ed., Halsted Press (John Wiley & Sons), New York, 1980.

[5.6] J. A. Hartigan, *Clustering Algorithms*, John Wiley & Sons, New York, 1975.

[5.7] G. N. Lance, W. T. Williams, "A General Theory of Classificatory Sorting Strategies: 1 Hierarchical System," *Comp. J.*, vol. 9, 1967, pp. 373-380.

[5.8] C. D. Himmel, G. S. May, "Advantages of Plasma Etch Modeling Using Neural Networks Over Statistical Techniques," *IEEE Trans. Semiconductor Manufacturing*, vol. 6, no. 2, May 1993, pp. 103-111.

[5.9] F. Nadi, A. Agogino, D. Hodges, "Use of Influence Diagrams and Neural Networks in Modeling Semiconductor Manufacturing Processes," *IEEE Trans. Semiconductor Manufacturing*, vol. 4, Feb. 1991, pp. 52-58.

[5.10] Stuttgart Neural Network Simulator, v. 3.0, Institute for Parallel and Distributed High Performance Systems, University of Stuttgart, 1990-94.

[5.11] R. P. Lippmann, "An Introduction to Computing with Neural Networks," *IEEE ASSP Mag.*, April, 1987.

Chapter 6

Wafer State Prediction

6.1 Introduction

The Wafer State Prediction Module uses empirical models based on real-time equipment data to predict the outcome of each wafer immediately after processing by each piece of equipment, reducing the need for costly and time-consuming wafer measurements. The prediction capability also allows the quality of the wafer to be known immediately after each process step, thereby obtaining important yield information to ensure that only wafers worth processing continue down the line. In high volume fabs, where several thousand wafers are processed each week, the chamber condition of the etchers changes over time. Therefore, it is crucial to verify that the equipment models survive these normal machine drifts. In this thesis, "prediction capability" refers to the proven ability of the model to describe the chamber, without further adjustments to the original model, after a significant number of wafers have caused the chamber to age.

As described in Chapter 4, the real-time data is decomposed into the long- and short-term components in the Fault Detection Module. In addition to being used to determine faulty behavior of the equipment, the long-term components are also used in the Wafer State Prediction Module to predict the wafer states immediately after the equipment has finished processing. In this way, it is possible to determine the effect of faults on the wafer quality. If the fault negatively impacts the wafer, the defective wafer can be eliminated from further processing, thereby saving the resources of the subsequent equipment. On the other hand, if the fault does not impact the wafer, the wafer can continue with the processing sequence.

Because the strength of this module relies on the models used for prediction, much of this chapter focuses on both the signals and the modeling methods used for prediction. To provide useful prediction capabilities, robust prediction models of the plasma etchers are required. The industry standard is to build models relating the input settings of the etchers to the output wafer state using methods such as response surface methodology (RSM) (Figure 6.1). Models using input settings, however, become unusable with time as the machine drifts with regular use, rendering them ineffective for prediction. Recently there has been much interest in using real-time tool data for modeling purposes. Elta *et al.* use information about the gas concentrations, the bias voltage, and the chamber pressure to model the wafer states for control purposes [6.1]. Anderson *et al* and Wangmaneerat showed that etch rate, selectivities, and uniformity can be well modelled with optical emission spectroscopy using partial least squares regression techniques [6.2][6.3]. While they have shown that models can be built relating real-time signals to wafer states, thus far they have not demonstrated actual prediction capability of the models spanning a significant number of wafers which will cause the machine to age. Work by Rietman and Lory show that neural network models can map the input settings and a few real-time signals,

including the induced DC bias and reflected RF power, to oxide thicknesses in the source and drain regions of CMOS devices [6.4].

In this chapter we show that successful wafer state prediction can be achieved by using a set of real-time data from key sensors inside the equipment. The signals used in this module reflect the RF components of the etcher, and were previously described in detail in Chapter 2. Because these real-time signals provide important information about the chamber state, we call the models built with real-time data *chamber state based (CSB)* models. This chapter shows that CSB models are effective for prediction because the real-time data reflect the actual state of the equipment as it changes over time.

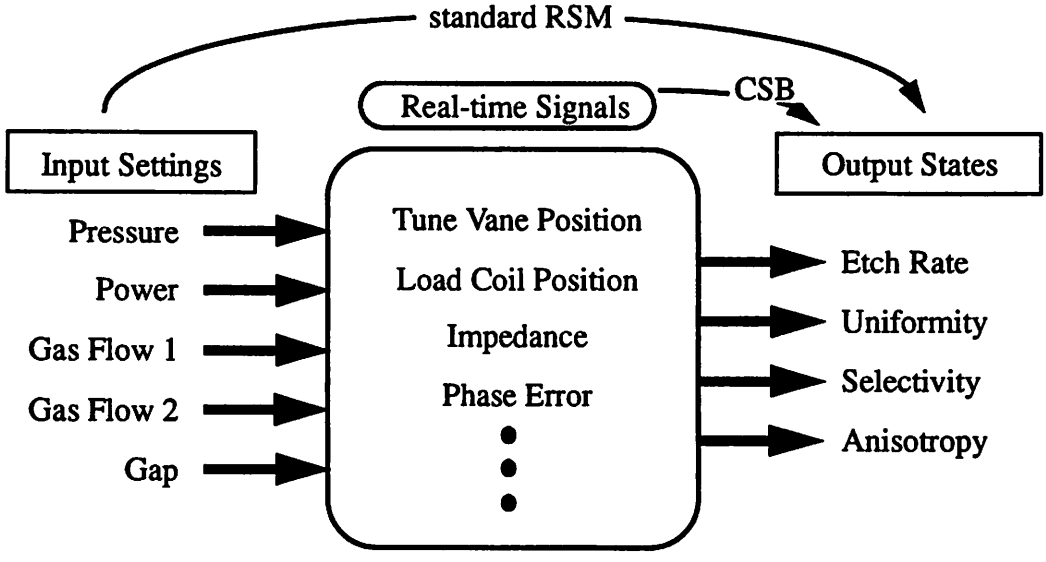


Figure 6.1 Wafer State Prediction: This paper shows that Chamber State Based (CSB) models, which map the real-time data to the output states, are effective for prediction even in the presence of equipment aging.

To develop the prediction models, two sets of experiments were conducted. During the experiments, both the input settings and the real-time data were simultaneously collected.

The wafer states of interest are the etch rates, selectivity, and uniformity. The first experiment, the Training Experiment, consists of a central composite design. The models using data from the Training Experiment relating either the input settings or chamber state data to the wafer states are called the training models. The second experiment, the Verification Experiment, was conducted several weeks later to determine the actual prediction capability of the training models. Two sets of models were developed. The first maps input settings directly to the output states, and will be referred to as standard RSM models. The second set of models are the CSB models, mapping the real-time signals to the output wafer states.

Three types of regression modeling methods, ordinary least squares regression, principal component regression, and partial least squares regression, for both sets of prediction models are explored. These regression models are also compared to models developed using feed-forward neural networks. The final prediction metric is determined by how well the training model predicts the wafer states of the Verification Experiment. This metric is a good measure of the actual predictive capability of the models because it is determined from runs performed much later in time and not included in model generation.

The goal of this chapter, then, is to show that real-time data collected while the machine is processing are well-suited for prediction of the wafer state. We also demonstrate the importance of the Verification Experiment and show how it affects the model prediction. The chapter begins with a brief recap of the real-time data used for the module, followed by a discussion of the methodology and models used to determine the wafer state prediction capability of the models. The modeling results based on the experiments described in Chapter 3 are then discussed.

6.2 Real-Time Data

As discussed in section 2.5, when the state of the chamber changes, the wafer-to-wafer variance of the real-time signals is much larger than the within-wafer variance. Thus, the input for the CSB models are the long-term components from the Fault Detection Module, which are the average values per signal over the duration of the main etch step of each wafer (after the native oxide breakthrough etch and before the overetch). In this chapter, the data used was collected during the Training and Verification Experiments described in Chapter 3. Approximately 30 points are collected per signal per wafer etch via LamStation, and 50 points via Real Power Monitor (RPM-1).

As illustrated in Figure 2.4 which shows the Load Impedance and RF Tune Vane Position for the duration of six wafers processed at the same input settings, the real-time signals change with the state of the machine even when the input settings remain fixed. A consequence is that the real-time data chosen for this analysis describe the actual equipment state more accurately than the input settings. Thus, the real-time data results in better predictive capability than the input settings, as is shown in section 6.8.

6.3 Wafer State Modelling Methods

This section outlines the basic advantages and disadvantages of four modeling methods, and discusses the prediction metric used to compare the prediction capability of the models. The first method under discussion is ordinary least squares regression. Since this method results in poor prediction capability when the modeling variables are correlated, other methods are investigated. Principal component regression and partial least squares regression can handle correlated data, and have the added advantage that they can reduce the dimensionality of the model. Simple feed-forward neural networks are also briefly discussed as an alternative modeling method.

6.3.1 Ordinary Least Squares Regression

The first regression method discussed is ordinary least squares regression (OLSR). The equation for the linear regression model is[1]

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\underset{\sim}{\beta}} \tag{6.1}$$

where $\hat{\mathbf{y}}$ $(n \times 1)$ is the prediction of the response $\mathbf{y}$, $\mathbf{X}$ $(n \times p)$ is the input matrix, and $\hat{\underset{\sim}{\beta}}$ is a $p \times 1$ vector of estimated model coefficients defined as

$$\hat{\underset{\sim}{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \tag{6.2}$$

provided that $(\mathbf{X}'\mathbf{X})$ is positive definite and therefore can be inverted. Throughout the chapter, n is the number of observations and p is the number of model parameters.

Prediction problems arise when the columns of $\mathbf{X}$ exhibit multicollinearity. The main idea is that high correlation in $\mathbf{X}$ leads to small eigenvalues in $\mathbf{X}'\mathbf{X}$, which can result in a high variance in both the estimate of the coefficients and the predicted responses. For example, let $\hat{y}_o = \mathbf{x}_o\hat{\underset{\sim}{\beta}}$ be a predicted value. The variance of this predicted value can be solved in terms of the eigenvalues w_j and eigenvectors $\mathbf{v}_j$ of $\mathbf{X}'\mathbf{X}$:

$$\text{var}(\hat{y}_o) = \text{var}\left(\mathbf{x}_o\hat{\underset{\sim}{\beta}}\right) = \mathbf{x}_o\text{cov}\left[\hat{\underset{\sim}{\beta}}, \hat{\underset{\sim}{\beta}}\right]\mathbf{x}'_o$$

$$= \sigma^2\mathbf{x}_o\sum_{j=1}^{p}\frac{\mathbf{v}_j\mathbf{v}_j'}{w_j}\mathbf{x}'_o = \sigma^2\sum_{j=1}^{p}\frac{\mathbf{x}_o\mathbf{v}_j\mathbf{v}'_j\mathbf{x}'_o}{w_j} \tag{6.3}$$

where $\text{cov}\left[\hat{\underset{\sim}{\beta}}, \hat{\underset{\sim}{\beta}}\right] = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$ when $\text{cov}[\mathbf{Y}, \mathbf{Y}] = \sigma^2\mathbf{I}_n$. Equation (6.3) shows that the variance of the predicted values depends on both the value of the eigenvalues and the

1. Bold face upper case letters denote matrices. Lower case bold face letters and Greek letters with an underscore ($\underset{\sim}{\ }$) denote column vectors. Scalars are denoted by lowercase letters. Transpose is denoted by (').

direction of the input $\mathbf{x}_o$. The variance will be large for small eigenvalues and large values of $\mathbf{x}_o\mathbf{v}_j$. The consequence of large variances in the predicted values is that the error in the prediction can potentially be huge. Thus, when the columns of **X** exhibit multicollinearity, both the estimates of the coefficients and the prediction capability of the model can be very poor.

6.3.2 Principal Component Regression

Principal component regression (PCR) addresses the problem of multicollinearity. When building models with real-time data, it is common to have large numbers of correlated input variables **X**. This number can easily escalate when interactions are included. For example, 13 main signals are collected, resulting in 90 model variables when the corresponding two-way interactions are included. Because many of the signals are correlated, not all 90 coefficients should (or can) be estimated independently.

Instead of artificially reducing the correlation among variables as in ridge regression, PCR transforms the input variables to a set of orthogonal variables. The transformed variables **Z**, known as the principal components (PC's), are linear combinations of the original variables. The value of these PC's are called the *scores*. The coefficients of the original variables, or *loadings*, are the eigenvectors **V** of $\mathbf{X}'\mathbf{X}$. The equation for the transformed variables **Z** is

$$\mathbf{Z} = (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}}')\,\mathbf{V} \tag{6.4}$$

where $\bar{\mathbf{x}}'$ is the vector of average values of each variable in **X** and **1** is a column vector of 1's. All or a subset of the PC's can be used as the input matrix for regression. Because the PC's are orthogonal, there are no multicollinearity problems, and standard least squares techniques can be employed. The resulting model is

$$\hat{\mathbf{y}} = \mathbf{Z}\hat{\underset{\sim}{\gamma}} \tag{6.5}$$

where $\hat{\underset{\sim}{\gamma}}$ is the estimate of the coefficients using the equation $\hat{\underset{\sim}{\gamma}} = (\mathbf{Z}'\mathbf{Z})^{-1}\mathbf{Z}'\mathbf{y}$.

Because much of the variability can be captured in a subset of the PC's, PCR reduces the dimensionality of the models to its most dominant factors. Assuming independence, the subset of statistically significant PC's in the model can be determined by calculating the *student-t* test for each of the coefficients. Only those PC's with statistically significant coefficients at a certain level (0.05 significance level is used in the examples of section 6.5) are retained in the model.

While PCR decreases the number of terms in the model, each model term still consists of a linear combination of input variables. Ideally, those input variables in **X** which do not significantly contribute to the model should be left out. When there are such large numbers of input variables, however, it is often very difficult to determine which of these simply add noise to the model and which are significant. An empirical method we developed to determine the "streamlined" models is to transform PCR model back to the input space of **X**. Assuming that the model is of the form shown in Equation (6.7) and using Equation (6.6) to substitute in for **Z**,

$$\hat{\mathbf{y}} = (\mathbf{X} - \mathbf{1}\bar{\mathbf{x}}')\,\mathbf{V}\hat{\underset{\sim}{\gamma}} = \mathbf{X}\mathbf{V}\hat{\underset{\sim}{\gamma}} - \mathbf{1}\bar{\mathbf{x}}\mathbf{V}\hat{\underset{\sim}{\gamma}} = \mathbf{X}\hat{\underset{\sim}{\beta}} - \mathbf{1}\bar{\mathbf{x}}\hat{\underset{\sim}{\beta}} \tag{6.6}$$

where $\hat{\underset{\sim}{\beta}} = \mathbf{V}\hat{\underset{\sim}{\gamma}}$. The general rule of thumb we found was to eliminate those input variables which have $\hat{\underset{\sim}{\beta}}$ values a magnitude or more smaller than the average of the few largest $\hat{\underset{\sim}{\beta}}$ values. This is similar to Cattell's "scree plots" used to determine the number of PC's which explain most of the variation in the original data [6.7]. We then regenerate the PCR model with the reduced set of input variables, using the *student-t* test to calculate the significance of the new PC's. Finally, we continue to reduce the input variable space as described above until the model prediction no longer improves. (An effective metric to

determine prediction is described in section 6.4.) This simple, yet effective empirical method handles large numbers of input variables easily.

6.3.3 Partial Least Squares Regression

The last statistical modeling technique under discussion is partial least squares regression (PLSR). This method is widely used in chemometrics, a field of chemistry that uses statistical methods for chemical data analysis [6.8][6.9]. Because the method is fairly new to statisticians, there has been much debate over its formal statistical properties and its relative predictive capabilities over OLSR, RR, and PCR. For example, it is often claimed that because the model generation uses information from both the input and output, PLSR results in better predictive models. This is not always the case, however, especially when the response data is noisy [6.10].

The general idea of the PLSR algorithm is similar to that of PCR. A reduced set of parameters that sufficiently describe the input data is found and then used as the regressors on **Y**. The notion of factor loadings and scores introduced in the context of PCR is also used in PLSR. Instead of one set of loadings as was the case in PCR, two sets are used in PLSR, one for the input matrix and another for the response. The algorithm for one response follows.

Let A_{max} be the maximum number of PLSR factors. At the start of the algorithm, A_{max} should be larger than anticipated to allow for unexpected factors. The following steps are then performed for each factor a = 1, 2,..., A_{max} [6.8][6.9][6.10].

1. Determine the loading weight vector $\hat{\mathbf{w}}_a$ using the model:

$$\mathbf{X}_{a-1} = \mathbf{y}_{a-1}\mathbf{w}'_a + \varepsilon.$$

where ε is the error. Use ordinary least squares to solve the following equation for $\hat{\mathbf{w}}_a$

$$\hat{w}'_a = \frac{y'_{a-1}X_{a-1}}{\|y_{a-1}\|} = y'_{a-1}X_{a-1}$$

since the length of y is 1 after scaling. Taking the transpose and scaling so that the length of $\hat{w}_a$ is 1,

$$\hat{w}_a = \frac{X'_{a-1}y_{a-1}}{\|X'_{a-1}y_{a-1}\|}.$$

The loadings $\hat{w}_a$ are orthonormal vectors which maximize the covariance between X_{a-1} and y_{a-1}. In other words, $\hat{W} = (\hat{w}_1, \hat{w}_2, \ldots, \hat{w}_A)$ relates the input and response, and will be used to calculate the response in the model.

2. Estimate the scores $\hat{t}_a$ by taking the projection of X_{a-1} onto $\hat{w}_a$:

$$X_{a-1} = t_a w'_a + \varepsilon.$$

Rewriting the model and solving for $\hat{t}_a$:

$$X'_{a-1} = w_a t'_a + \varepsilon$$

$$\hat{t}'_a = \frac{\hat{w}'_a X'_{a-1}}{\|\hat{w}_a\|} = \hat{w}'_a X'_{a-1}$$

Taking the transpose:

$$\hat{t}_a = X_{a-1}\hat{w}_a.$$

$\hat{t}_a$ indicates how much of the response is correlated with the input data, and $\hat{T} = (\hat{t}_1, \hat{t}_2, \ldots, \hat{t}_A)$ is the reduced set of orthogonal scores that are used as regressors for **Y**. Orthogonal vectors are necessary to deal with the problem of multicollinearity.

3. Estimate the input loadings $\hat{p}_a$ using the model:

$$X_{a-1} = \hat{t}_a p'_a + \varepsilon.$$

Solving for $\hat{\mathbf{p}}_a$:

$$\hat{\mathbf{p}}'_a = \frac{\hat{\mathbf{t}}'_a \mathbf{X}_{a-1}}{\|\hat{\mathbf{t}}_a\|}$$

and thus taking the transpose,

$$\hat{\mathbf{p}}_a = \frac{\mathbf{X}'_{a-1} \hat{\mathbf{t}}_a}{\|\hat{\mathbf{t}}_a\|}.$$

$\hat{\mathbf{P}} = (\hat{\mathbf{p}}_1, \hat{\mathbf{p}}_2, \ldots, \hat{\mathbf{p}}_A)$ is similar to the eigenvector matrix $\mathbf{V}$ in PCR, in that it consists of the loadings for the input. Although $\hat{\mathbf{P}}$ is chosen to ensure that the $\hat{\mathbf{t}}_a$ vectors are orthogonal, the $\hat{\mathbf{p}}_a$ vectors are generally not orthogonal. Unlike the loadings in PCA, the first $\hat{\mathbf{p}}_a$ vector does not explain the maximum variance in the input matrix; rather, it explains as much variance as possible while correlating with the response.

4. Estimate the response loadings $\hat{q}_a$ using the model:

$$\mathbf{y}_{a-1} = \hat{\mathbf{t}}_a q_a + \gamma .$$

Solving for $\hat{q}_a$:

$$\hat{q}_a = \frac{\mathbf{y}'_{a-1} \hat{\mathbf{t}}_a}{\|\hat{\mathbf{t}}_a\|}.$$

Thus, $\hat{\mathbf{Q}} = (\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_A)$ is the additional loading term which brings the response into the model. It relates the score $\hat{\mathbf{t}}_a$ to the response, minimizing the residual sum of squares of the response. Note that $\hat{q}_a$ are scalars since this model is for one response.

5. Create the new residuals $\hat{\varepsilon}$ and $\hat{\mathbf{F}}$ by subtracting the estimated values found in the previous steps from the actual values:

$$\hat{\varepsilon} = \mathbf{X}_{a-1} - \hat{\mathbf{t}}_a \hat{\mathbf{p}}'_a$$

$$\hat{\mathbf{F}} = \mathbf{y}_{a-1} - \hat{\mathbf{t}}_a \hat{q}_a .$$

The product $\hat{\mathbf{t}}_a\hat{\mathbf{p}}'_a$ estimates the input matrix, while the product $\hat{\mathbf{t}}_a\hat{\mathbf{q}}_a$ estimates the response matrix. Replace $\mathbf{X}_{a-1}$ and $\mathbf{y}_{a-1}$ by the new residuals and increment a:

$$\mathbf{X}_a = \hat{\varepsilon}, \mathbf{y}_a = \hat{\mathbf{F}}, \text{and } a = a + 1.$$

Go back to Step 1.

6. Once the number (A) of valid PLSR factors is determined, the estimate of the coefficients to be used in the prediction model $\hat{\mathbf{y}} = \mathbf{1}\hat{\beta}_o + \mathbf{X}\hat{\underset{\sim}{\beta}}$ are

$$\hat{\underset{\sim}{\beta}} = \hat{\mathbf{W}}(\hat{\mathbf{P}}'\hat{\mathbf{W}})^{-1}\hat{\mathbf{q}} \text{ and } \hat{\beta}_o = \bar{\mathbf{y}} - \bar{\mathbf{x}}'\hat{\underset{\sim}{\beta}}. \tag{6.7}$$

Using Equation (6.9) as an estimate of the coefficients, the same type of "streamlining" method described for PCR to reduce the number of input parameters can also be applied to PLSR.

Like PCR, the scores of PLSR are not scale invariant. For example, suppose two variables are measured in meters and kilograms, and the desired scores are to be expressed in centimeters and grams. One way to achieve this is to first transform the variables to the desired units of centimeters and grams, and then carry out the PLS analysis. The second method is to first perform the PLS analysis in meters and kilograms and then multiply the elements of the relevant scores by the proper scaling factors (100 and 1000, respectively). The two methods do not result in the same solutions since the scores of a random vector are not scale invariant. Thus, as was suggested for PCR, it is sometimes useful to standardize the data so that all variables are equally weighted in the analysis. For computational purposes, centering and scaling reduces round off and overflow problems. If it is known, however, that variables with small values (in magnitude) are less important than those with larger values, scaling is inappropriate.

6.3.4 Feed-Forward Error Backward Propagation Neural Networks

The last modeling method investigated is neural networks, which has emerged as an effective modeling method for semiconductor equipment [6.4][6.11][6.12][6.13]. Neural networks are useful for modeling complex relationships, such as the plasma etching process. Furthermore, the form of the models is derived from the actual data, and not set *a priori* as is done for regression. Neural networks, however, do not provide information about the physics of the processes [6.13][6.4][6.12].

The network selected for this analysis is the feed-forward error back propagation (FFEBP) algorithm, which was described in section 5.4.2. In this application, one hidden layer was used, making a total of three layers in the network. Several different structures were investigated, and the final structure chosen was the one which resulted in the smallest error. The connections are between the input nodes and the hidden nodes, and between the hidden nodes and the output nodes. No bias was applied to the first layer. The output function for the remaining layers is the "squashing" activation function of the form $f(x) = \frac{1}{1+e^{-x}}$, where x is the sum of the weighted outputs of the nodes preceding this particular node.

As in Chapter 5, the Stuttgart Neural Network Simulator (SNNS) was used to train and simulate the neural networks [6.14]. The network learns the relationship between the input and output patterns as it undergoes learning iterations. To determine when to stop training, the neural network model was applied to the verification data set. Training stopped when this testing set achieved its lowest error. This is a usual practice to eliminate over-training, which results in decreased generalization capability of the network model.

6.4 Testing the Prediction Capability of the Models

This section describes the methodology used to determine the prediction capability of the models. As stated in section 6.1, two sets of experiments were conducted, the first for model generation and the second for model verification. These are described in detail in sections 3.2.2 and 3.2.3. It is important to note that the two experiments were conducted several weeks apart, and that between the experiments the equipment underwent normal use and maintenance. The Verification Experiment is used to determine if the training models can withstand small changes in the equipment that occur with time.

The often neglected verification stage is one of the most important in prediction model building. In many modeling situations, the assumption is made that if the model has a good fit (for example, a high adjusted R^2 and statistically significant terms), the model can be used well for prediction. Unfortunately, this is not the case for plasma etchers on a production line. Because the machines go through regular maintenance and may drift with use, the model with the best fit based on one experiment conducted in a short time frame may not capture these changes in the machine. The model may also be too specific for the particular runs. These combined deficiencies result in unsatisfactory predictive capability. The verification experiment is designed to determine the best predictive model which takes into account normal equipment changes.

The prediction metric determining the best model is based on how well the training model predicts the verification wafers. Because the verification data is not included during model generation, the true prediction capability of the models can be gauged. The metric used is the standard error of prediction (SEP), where $\mathbf{Y}_i$ is the *i*th observation, $\hat{\mathbf{Y}}_i$ is the predicted value of the *i*th point, and *n* is the number of observations in the experiment:

$$SEP = \sqrt{\frac{\sum_{i=1}^{n} \left(Y_i - \hat{Y}_i \right)^2}{n-1}}. \quad (6.8)$$

Essentially, the SEP metric measures the spread of the difference between the predicted and actual values, called the residuals. The examples shown in section 6.5 rate the different models according to their normalized SEP metrics. To determine whether two SEP values are statistically significantly different, we employ the standard F-test of hypothesis, assuming that the residuals are approximately normally distributed. The null hypothesis is that the squared SEPs are equal, while the test statistic is the ratio of the squared SEPs.

6.5 Polysilicon Etch Rate Modeling Results

In this section, a detailed discussion is given for the CSB polysilicon etch rate model analysis using the four different modeling techniques described in the previous sections. Since the same type of analysis is conducted for the other wafer states, those results are summarized in the following two sections.

6.5.1 Ordinary Least Squares and Ridge Regression

Using standard methods outlined in section 6.3.1, the first prediction modeling method investigated is OLSR. Due to restrictions on the degrees of freedom, only the main effects and interactions of the RPM-1 data was used to build the model. The data was scaled to have zero mean and unity variance after the interactions were created. Backward stepwise regression was employed to choose the significant terms in the model at the 0.05 significance level.

The results of the regression models are summarized in the ANOVA table in Table 6.1. The table shows the degrees of freedom (d.f.), the sum of squares (SS), and mean sum of

squares (MS) of both the regression and residual. The test statistic, which has an F-distribution, is calculated as the regression MS divided by the residual MS. Also listed is the adjusted R^2 statistic, which takes into account the number of terms used in the model.

Table 6.1 ANOVA Table for OLSR Model of Polysilicon Etch Rate

Source	d.f.	SS	MS	F
Regression	15	3.06×10^6	2.04×10^5	24.9
Residual	11	9.01×10^4	8.19×10^3	
adj. $R^2 = 0.930$				

Despite a high adjusted R^2 statistic of 0.93 the model fails as a prediction /min model, with a verification SEP metric of 1138 Å (22.2% when normalized by the average polysilicon etch rate of the Verification Experiment). Depending on the input variables used in the OLSR models, the prediction error can become quite large, even when all the terms in the model are statistically significant at the 0.05 level. For example, when the main variables and some interactions from the Lam Station data are included in the model, the SEP metric is 6767 Å/min (132%), despite that all terms are statistically significant and the adjusted R^2 statistic is 0.999. This example illustrates the importance of evaluating the models against data not used to build the model because the usual metrics such as the adjusted R^2 statistic can be misleading. This is because the correlations among the input variables lead to high correlations among the estimated coefficients, resulting in unstable prediction. Thus, standard OLSR techniques are not satisfactory for predicting polysilicon etch rate based on highly correlated real-time signals. Similar results are found for the other wafer states of interest.

6.5.2 Principal Component Regression

Because of their ability to handle large amounts of correlated data, PCR and PLSR methods are much better suited to handle real-time data. The PCR model has an input

variable space consisting of all 13 real-time signals with their corresponding two-way interactions, making a total of 90 parameters. In this analysis, the 13 signals were scaled to have zero mean and unity variance before forming the interactions. The PC's which explain 99% of the variance were then included in model generation. Variable selection, using the *student-t* test at the 0.05 significance level, resulted in a model with an intercept and three PC's, resulting in a verification SEP metric of 704 Å/min (10.7%). This is a tremendous improvement over the OLSR model.

The coefficients of this PCR model with 90 input variables can be transformed from PCA space back to the input space **X** using the equation $\hat{\underset{\sim}{\beta}} = \mathbf{V}\hat{\gamma}$, derived in section 6.3.3. None of the coefficients are orders of magnitudes greater than the others, which is a consequence of scaling the input variables of **X**. Several of the coefficients have values that are close to zero, however, and may not be important to the model. Following the algorithm outlined in section 6.3.3 to "streamline" the models, thirteen variables corresponding to those columns of **X** with small (< 4.0) estimated coefficients were eliminated from the input space of the original PCR model, resulting in 77 terms in **X**. The results from using the eigenvalues that explain 99% of the variation were: adjusted $R^2 = 0.70$, Mallow's Cp = 7, and SEP = 688 Å/min (13%), which is a slight improvement over the previous model. An additional 25 input variables were eliminated on the next "streamlining" iteration, which resulted in the best prediction. The SEP is 496 Å/min (9.7%). Five terms are in the model, one intercept term and four PC's corresponding to the 3rd, 5th, 6th, and 8th largest eigenvalues. The ANOVA table listed below in Table 6.2 summarizes the model.

Table 6.2 ANOVA Table for PCR Model of Polysilicon Etch Rate

Source	d.f.	SS	MS	F
Regression	4	2.23×10^6	5.58×10^5	13.27
Residual	22	9.25×10^5	4.20×10^4	
adj. $R^2 = 0.640$				

The plot of predicted vs. actual polysilicon etch rate in units of Å/min is shown in Figure 6.2. If the models were perfect, all the points would lie on the $y=x$ line. The spread of the training and verification data is about equal, and the model shows a full range of output coverage. Further reductions in the input space did not lead to better prediction.

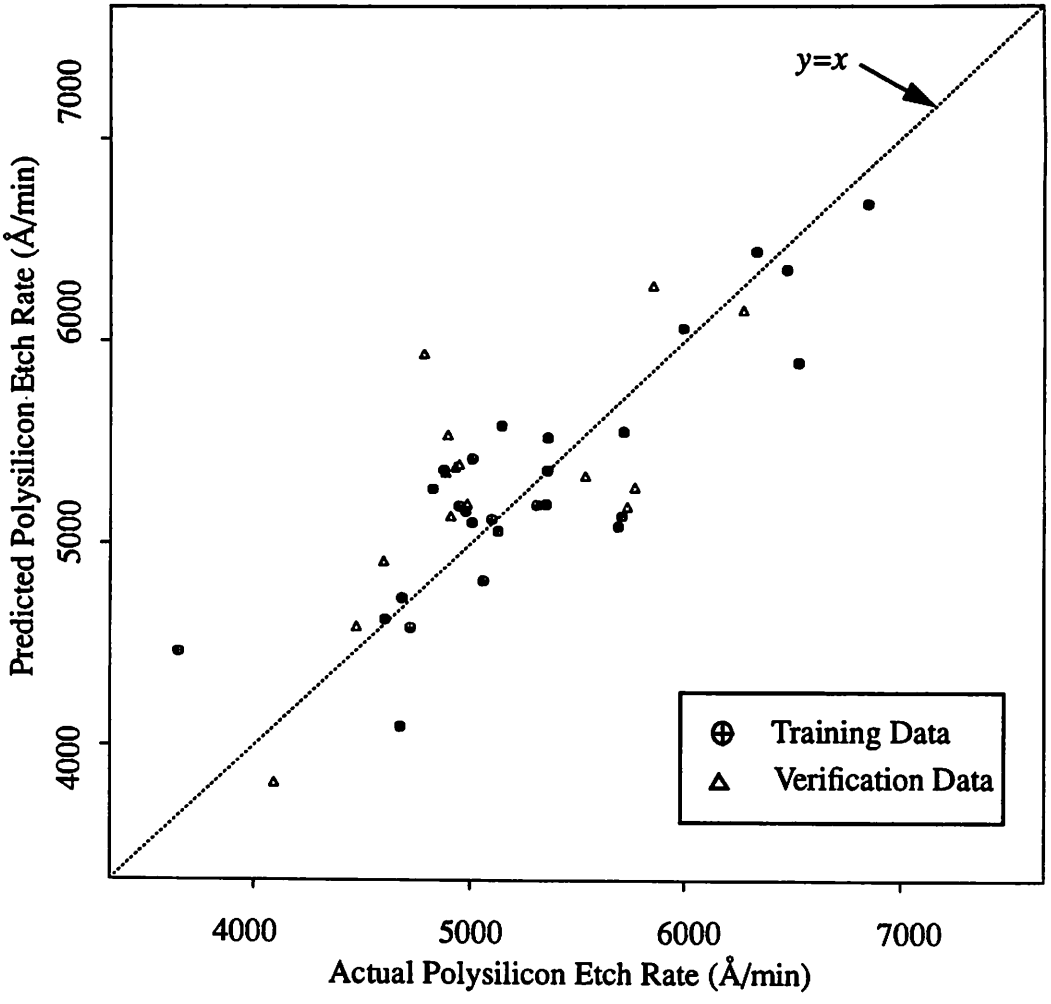


Figure 6.2 Predicted versus actual polysilicon etch rate plot of the best predictive model found via principal component regression. SEP = 9.7%.

6.5.3 Partial Least Squares Regression

Like PCR, partial least squares regression also reduces the number of terms in the model. Although the transformed variables are not orthogonal, they relate the response to

the inputs. As in the previous section, the first model is built using input matrix **X**, which consists of the 13 main effects and all the two-way interactions, making a total of 90 variables in **X**. The 13 variables were scaled before forming the interactions. Models ranging from one to 21 terms were built using PLSR. Like PCR, the prediction error does not escalate as severely for overfitted models as it does for OLS regression.

The model with four terms has an SEP value of 549 Å/min (10.7%). When the models

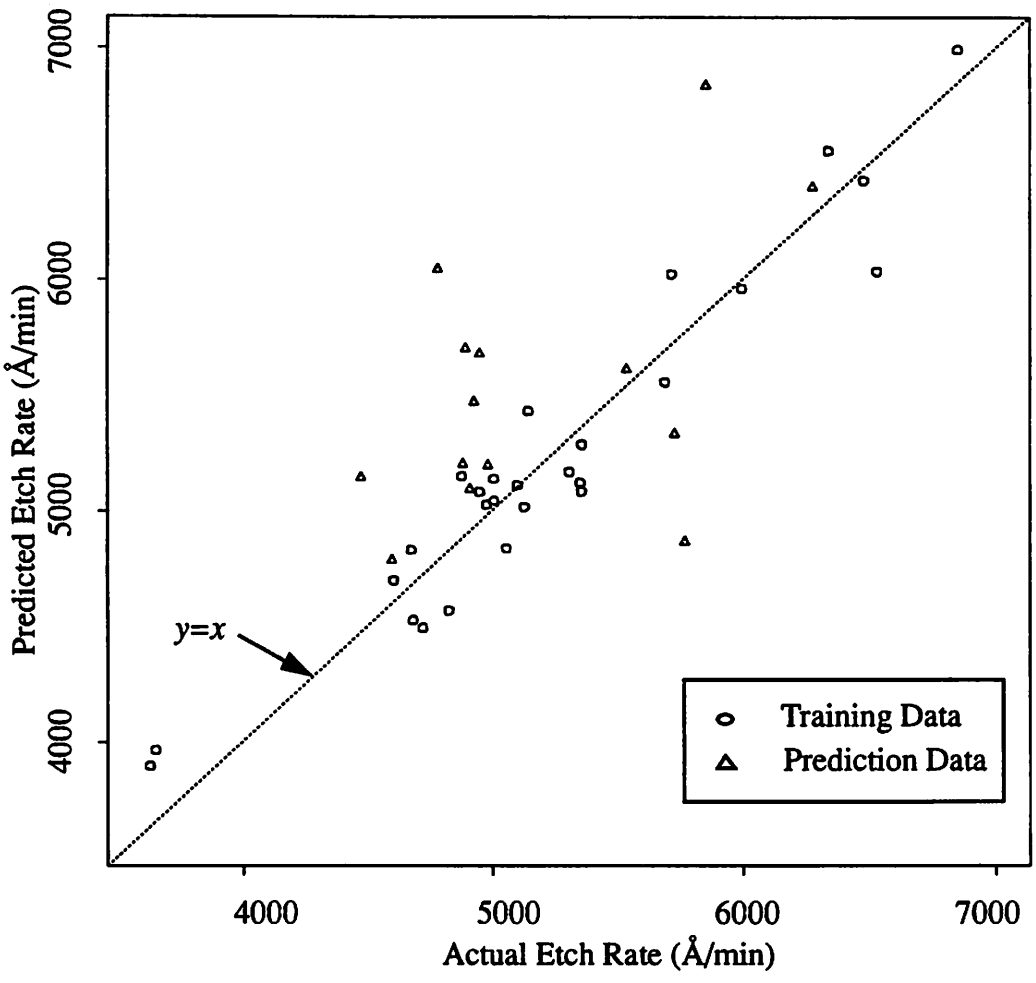


Figure 6.3 Predicted versus actual polysilicon etch rate plot of the best predictive model found via partial least squares regression. SEP = 10.5%.

were "streamlined," the SEP dropped slightly to 539 Å/min (10.5%). This model is shown

in Figure 6.3. The main benefit in using PCR and PLSR models is that they are much less sensitive to overfitting and resulted in more stable models than OLSR.

6.5.4 Feed-Forward Error Backward Propagation Neural Networks

The predictive capability of the FFEBPNN model is about the same as the PCR model, with a slightly higher SEP value, 500 Å/min (9.8%). The input and output patterns of the neural network model were first scaled to lie between 0 and 1. The best FFEBPNN model structure was of the form 6-5-1, where the six input nodes correspond to the 6 signals collected via the RPM-1, and the 3 output nodes were the etch rates of polysilicon, oxide, and photoresist. Learning was fast, with only 100 iterations needed to attain the best FFEBPNN model for polysilicon etch rate. As in the previous statistical models, 27 runs were used for training, and 15 were used for testing of the model.

6.5.5 Comparison of the Models

Table 6.3 gives a comparison of all four modeling methods, in terms of the number of variables in **X** in each model, the verification SEP, and the normalized SEP. The F-test shows that the SEP values of the PCR, PLSR, and FFEBPNN models are statistically better than that of the OLSR model at the 0.05 level. Furthermore, PCR, PLSR, and FFEBPNN methods applied to the real-time data are equally good for polysilicon etch rate prediction since their SEP values can not be distinquished from one another.

Table 6.3 Summary of CSB Models For Polysilicon Etch Rate

Training Model Description		Verification	
Model type	# of Input Variables in **X**	SEP (Å/min)	normalized SEP
OLSR	16	1138	22.2%
PCR	52	496	9.7%
PLSR	63	540	10.5%

Table 6.3 Summary of CSB Models For Polysilicon Etch Rate

Training Model Description		Verification	
Model type	# of Input Variables in **X**	SEP (Å/min)	normalized SEP
FFEBPNN	90	500	9.8%

6.6 Selectivity Models

Due to the small ranges of selectivities across the design space, models are created for the individual etch rates of gate oxide and photoresist instead of modeling the selectivities. Since the analysis for these etch rates is similar to that of polysilicon, only a summary of the models is given here.

Like the case for polysilicon etch rate, the PCR, PLSR, and FFEBPNN models for oxide etch rate resulted in statistically significantly better prediction than the OLSR model, as shown in Table 6.4. Once again, the PCR and PLSR models can not be distinguished. The best PCR model was built with unscaled data, while the PLSR used scaled (mean 0, variance 1) data. The neural network models, however, resulted in statistically significantly worse prediction than the regression models for the oxide etch rate at the 0.05 level. Several structures were tested, and the best FFEBPNN model had the 6-5-3 structure, where once again, the inputs corresponded to the RPM signals, and the data was scaled to lie between 0 and 1.

Table 6.4 Summary of CSB Models For Oxide Etch Rate

Training Model Description		Verification	
Model type	# of Input Variables in **X**	SEP (Å/min)	normalized SEP
OLSR	16	216	52.7%
PCR	39	39	6.1%

Table 6.4 Summary of CSB Models For Oxide Etch Rate

Training Model Description		Verification	
Model type	# of Input Variables in X	SEP (Å/min)	normalized SEP
PLSR	35	31	7.5%
FFEBPNN	6	70	16.9%

While the OLSR model resulted in the worst prediction error for the photoresist etch rate, the PCR model and the neural network model, using all 13 signals as input, 5 nodes in the hidden layer, and 3 output nodes, resulted in the best prediction capability. As in the models for gate oxide, the best PCR model was built with unscaled data, the PLSR used scaled data, and the neural network inputs were scaled to lie between 0 and 1. The model results are listed in Table 6.5.

Table 6.5 Summary of CSB Models For Photoresist Etch Rate

Training Model Description		Verification	
Model type	# of Input Variables in X	SEP (Å/min)	normalized SEP
OLSR	13	901	29.1%
PCR	39	148	4.8%
PLSR	90	280	9.0%
FFEBPNN	13	117	3.8%

6.7 Polysilicon Uniformity Models

As shown in Table 10, none of the models for polysilicon uniformity are useful for prediction. The models for uniformity may be improved if additional signals are included in the analysis. The real-time signals used in this paper are essentially an average of the specific etch processes, and as such, give no spatial information about the chamber. There-

fore, the real-time signals used in this paper are not ideal to provide meaningful uniformity measurements. A set of chamber state signals which have shown promise for uniformity prediction is spatially resolved optical emission spectroscopy [6.3]. PLSR, designed specifically to model OES data, has been shown to be effective for training uniformity models based on OES data [6.3], and may prove to be the modeling method of choice. Thus far, however, the predictive capability has not been tested for OES data. Another new sensor showing promise is the full wafer interferometry system, which has the potential of calculating the etch rate of points across the entire wafer. This method, however, requires hardware changes to many present etching systems since it relies on a top view of the wafer [6.15].

Table 6.6 Summary of CSB Models For Polysilicon Uniformity

Training Model Description		Verification	
Model type	# of Input Variables in **X**	SEP (Å/min)	normalized SEP
OLSR	18	35.5	546%
PCR	90	4.8	73.8%
PLSR	90	4.34	66.8%
FFEBPNN	13	8.2	126%

6.8 Comparison of Chamber State Based and Response Surface Methodology Models

In this section, the prediction capability of the CSB and standard RSM models are compared. All four modeling methods described in the previous sections were investigated, and the model with the smallest SEP for each set of inputs was chosen for comparison. In all cases, the CSB models built with the real-time data are have approximately the same prediction capability as the models built with input settings.

Although these examples show the prediction capability to be similar, we expect that the CSB models will prove to be superior when more time has elapsed and the equipment has drifted from its original setpoint because unlike the fixed input settings, the real-time signals change with the state of the machine. Figure 6.6 compares the modeling results of six centerpoint wafers. The standard RSM model built with the fixed input settings predicts a constant etch rate, while the real-time model adjusts the prediction as a result of small changes in the machine state. Thus, we surmise that models built using real-time data may predict etch rates with more accuracy than those built with input settings in the presence of machine drift. One reason this is not seen with the present test case is because the range of the Verification Experiment was simply not large enough to stress the models. Only one variable was altered at a time, and the range was quite small. As a result, the range of the etch rate was not much greater than the natural variation of the centerpoint data. A better experiment to perform should include changes to more than one input setting at a time. In addition, the range of the settings should have a larger range.

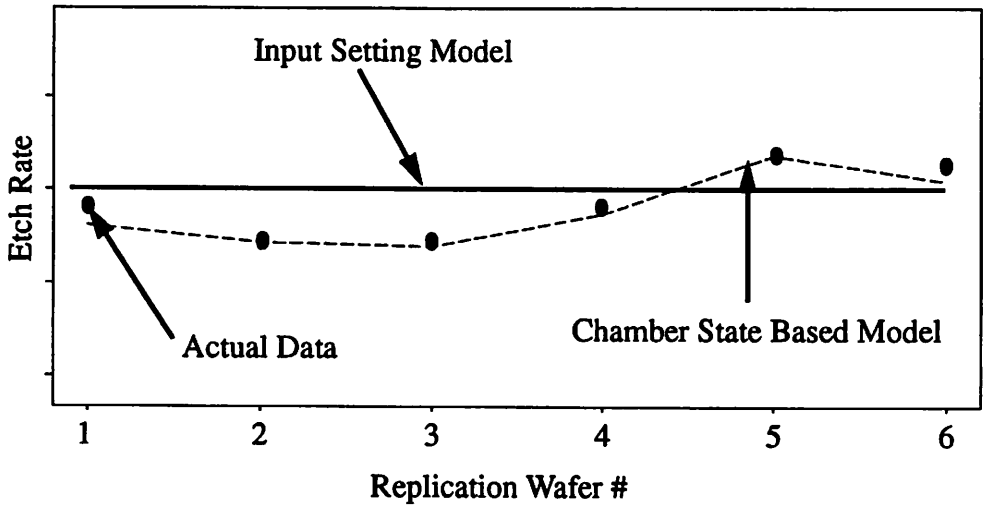


Figure 6.4 Comparison of the model built with input settings versus the chamber state based model built with real-time signals.

6.9 Wafer State Prediction Module Summary

Plasma etch models using real-time equipment signals lead to excellent prediction models for etch rates (and thus selectivities as well). Additional signals may be needed for more accurate prediction of wafer uniformity. Four different modeling techniques, ordinary least squares regression (OLSR), principal component regression (PCR), partial least squares regression (PLSR), and feed-forward error back propagation neural networks (FFEBPNN) were implemented. OLSR can not be used successfully for wafer state prediction because the real-time signals are highly correlated, resulting in severe instabilities in the predicted values. PCR, PLSR, and FFEBPNN are well-suited to handle large numbers of correlated input variables. The prediction capability was verified on data collected several weeks after the initial experiment. Because real-time data reflects the actual chamber state of the equipment, models based on this real-time data, called chamber state based (CSB) models, can be used effectively for prediction of etch rates.

The Wafer State Prediction Module presented is especially powerful because it uses non-invasive real-time signals collected automatically from the tool while the wafer is processing. Since the wafer parameters are predicted immediately after the wafer has finished processing in the machine, important yield information is obtained on a run-to-run basis, making it possible to ensure that only wafers worth processing continue down the line. The real-time signals can also be used to qualify equipment to determine if the machine is operating properly.

References for Chapter 6

[6.1] M. Elta, J. S. Freudenberg, *et al*, "Applications of Control to Semiconductor Manufacturing: Reactive Ion Etching," *Proceedings of the American Control Conference*, San Francisco, CA, June 1992, pp. 2990-2997.

[6.2] B. Wangmaneerat, "Chemometric Data Analysis of Infared and Visible Emission Spectral Data for Quantitative Property Determinations," Doctoral Thesis, University of New Mexico, May 1992.

[6.3] H. M. Anderson, M. P. Splichal, "An Integrated System of Optical Sensors for Plasma Modeling and Plasma Process Control," *Proc. of SPIE*, vol. 2091, Monterey, CA, Sept. 27-29, 1993, pp. 333-334.

[6.4] E. A. Rietman, E. R. Lory, "Use of Neural Networks in Modeling Semiconductor Manufacturing Processes: An Example for Plasma Etch Modelling," IEEE *Trans. Semiconductor Manufacturing*, vol. 6, no. 4, Nov. 1993, pp. 343-347.

[6.5] A. E. Hoerl, R. W. Kennard, "Ridge Regression. Biased Estimation for Non-Orthogonal Problems," *Technometrics*, vol. 12, pp. 55-67, 1970.

[6.6] A. E. Hoerl, R. W. Kennard, "Ridge Regression. Applications to Non-Orthogonal Problems," *Technometrics*, vol. 12, pp. 69-82, 1970.

[6.7] R. B. Cattell, "The Scree Test for the Number of Factors," *Multivariate Behav. Res.*, vol. 1, pp. 245-276, 1966.

[6.8] I. E. Frank, J. H. Friedman, "A Statistical View of Some Chemometrics Regression Tools," *Technometrics*, vol. 35, no. 2, pp. 109-135, 1993.

[6.9] H. Wold, "Estimation of Principal Components and Related Models by Iterative Least Squares," *Multivariate Analysis*, ed. P. R. Krishnaiah, Academic, pp. 391-420, 1966.

[6.10] H. Martens, T. Naes, *Multivariate Calibration*, Wiley, 1989.

[6.11] B. Kim, G. S. May, "An Optimal Neural Network Process Model for Plasma Etching," *IEEE Trans. Semiconductor Manufacturing*, vol. 7, no. 1, Feb. 1994, pp. 12-21.

[6.12] C. D. Himmel, G. S. May, "Advantages of Plasma Etch Modeling Using Neural Networks Over Statistical Techniques," *IEEE Trans. Semiconductor Manufacturing*, vol. 6, no. 2, May 1993, pp. 103-111.

[6.13] F. Nadi, A. Agogino, D. Hodges, "Use of Influence Diagrams and Neural Networks in Modeling Semiconductor Manufacturing Processes," *IEEE Trans. Semiconductor Manufacturing*, vol. 4, Feb. 1991, pp. 52-58.

[6.14] Stuttgart Neural Network Simulator, v. 3.0, Institute for Parallel and Distributed High Performance Systems, University of Stuttgart, 1990-94.

[6.15] D. S. Boning, J. L. Claman, K. S. Wong, T. J. Dalton, H. H. Sawin, "Plasma Etch Endpoint via Interferometric Imaging," *Proceedings of the American Control Conference*, June 1994, pp. 897-901.

Chapter 7

Conclusions

7.1 Thesis Summary

This thesis presented a system which detects equipment malfunctions in real-time, assigns a cause to the problem, and finally determines the impact of the fault on the final wafer characteristics. Key to the success of this Equipment Analysis and Wafer State Prediction System is the data used in the analysis, namely non-invasive real-time signals which can be automatically collected from the equipment while the wafers are processed. For plasma etchers in particular, we have isolated a few key signals from two collection systems, the Brookside LamStation software and the Comdel Real Power Monitor. These signals have shown to be much more sensitive to the actual state of the chamber than the input settings of the machine [7.1].

The real-time signals are used in each of the system's three modules, the Fault Detection, Fault Diagnosis, and Wafer State Prediction Modules. We have improved the Fault

Detection Module, which uses time series modeling and Hotelling's T^2 statistic to detect equipment malfunctions [7.2]. Algorithms have also been investigated to extend the module to include multiple recipes without retraining the system for each recipe. The Fault Detection Module has been successfully applied to many single wafer etching systems.

Two methods have been studied to perform fault diagnosis and prognosis for the Diagnosis Module. The first, using discriminant analysis, is easy to train and requires no knowledge of the equipment [7.3]. It is scale sensitive, however, and may require numerous training runs. The second method, using clustering and neural network techniques, is based on heuristics. It is scale invariant and can be used for prognosis as well as diagnosis. Examples showing effective equipment diagnosis using both methods have been demonstrated.

The third module, Wafer State Prediction Module, assesses the quality of the wafer once it has completed processing. This module relies on accurate models of the critical output wafer states. Four modelling techniques were evaluated, ordinary least squares regression (OLSR), principal component regression (PCR), partial least squares regression (PLSR), and feed-forward error back propagation neural networks (FFEBPNN). The prediction capability of the models was measured by using data collected several weeks after the training set, and not used to build the models. It was shown that the techniques which handle highly correlated data, namely PCR, PLSR, and NN, result in more stable models than OLSR. Chamber state based (CSB) models, which use the real-time data, are effective in predicting the output wafer states [7.1].

Although the examples developed in this thesis are based on data collected from single wafer plasma etchers using the LamStation and Comdel RPM-1 sensors, the methodology presented is general and can be applied to other types of equipment and sensor data. For example, data collected via optical emission spectroscopy can be used in exactly the same

manner. A current research area is to determine the sensor data set which precisely describes the chamber state. At present we have found data collected from the Brookside LamStation software and the Comdel Real Power Monitor to be sufficient to show the power of this class of real-time equipment data. This system can also be applied to other semiconductor equipment. The most straight-forward extension is most likely to chemical vapor deposition furnaces, cluster tools, and multi-chamber systems.

7.2 Future Directions

7.2.1 Short-Term

A few areas in each module require more study and research. First, the ideas to use ARIMAX models to extend the Fault Detection Module to several different recipes must be verified. A more difficult problem is to include effects of different wafer loading of the wafers. This is important for fabs which produce a large mix of products. Second, the staged clustering method in the Fault Diagnosis Module potentially requires different classification heuristics for each type of etcher. It is also unclear how this algorithm extends to multiple faults. Third, the uniformity models in the Wafer State Prediction Module are unsatisfactory for production use. Other sets of data, such as spatially resolved optical emission spectroscopy, may result in much more accurate uniformity models [7.4][7.5]. Also promising is the full wafer interferometric imaging system, which extracts the etch rates across an entire wafer during processing [7.6].

7.2.2 Long-Term

The Fault Detection Module can presently be run as a stand-alone package, or within the existing Berkeley Computer Aided Manufacturing (BCAM) Framework [7.7], which has capabilities to perform recipe generation and run-to-run control. In the future, the Fault Diagnosis Module can fit into the BCAM Dempster-Shafer Evidential Reasoning

Diagnostic Framework, which collects evidence from three process stages: equipment maintenance history before the wafer enters the equipment, real-time sensor data while the wafer is processing, and the in-line measurements when the wafer leaves the equipment [7.8]. To take advantage of the complete diagnostic method, however, this fault classification should be translated into a form of numerical belief about malfunctions. The real-time equipment faults can be classified into different sets, each with a certain misclassification rate. Therefore, we can determine the probability that a certain equipment fault has occurred. This probability value can be used as *support* within the BCAM evidential reasoning system. This system will effectively combine this support with the evidence collected during the maintenance and in-line diagnostic phases in order to produce a valid, ranked fault list. The challenge in this translation is the design of an efficient experiment to determine baseline behavior and fault categories, and the creation of evidence combination rules that effectively take advantage of the real-time information [7.3].

A consequence of the prediction capability of the CSB models in the Wafer State Prediction Module is that inexpensive run-to-run control is possible. In the absence of reliable wafer state prediction, work in run-to-run control specifically for plasma etching has included the use of *in-situ* sensors such as spectral ellipsometry [7.9][7.10]. Wafer state prediction will allow a run-to-run control scheme of plasma etch equipment that will bring specified output parameters back to their target value in the case of equipment drift.

References for Chapter 7

[7.1] S. F. Lee and C. J. Spanos, "Prediction of Wafer State After Plasma Processing Using Real-Time Tool Signals," submitted to *IEEE Trans. Semiconductor Manufacturing*, Aug. 1994.

[7.2] S. F. Lee, E. D. Boskin, H. C. Liu, E. Wen, C. J. Spanos, "RTSPC: A Software Utility for Real-Time SPC and Tool Data Analysis," to appear in *IEEE Trans. Semiconductor Manufacturing*.

[7.3] C. J. Spanos, S. Leang, S. F. Lee, "A Control and Diagnosis Scheme for Semiconductor Manufacturing," *American Control Conference*, vol. 3, June 1993, pp. 3008-3012.

[7.4] B. Wangmaneerat, Chemometric Data Analysis of Infared and Visible Emission Spectral Data for Quantitative Property Determinations, Doctoral Thesis, University of New Mexico, May 1992.

[7.5] Anderson, H. M., Splichal, M. P, "An Integrated System of Optical Sensors for Plasma Modeling and Plasma Process Control," *Proceedings of SPIE*, vol. 2091, Monterey, CA, Sept. 27-29, 1993, pp. 333-334.

[7.6] D. S. Boning, J. L. Claman, K. S. Wong, T. J. Dalton, H. H. Sawin, "Plasma Etch Endpoint via Interferometric Imaging," *Proceedings of the American Control Conference*, June 1994, pp. 897-901.

[7.7] *BCAM 3.1 User's Manual*, version 3.1, Berkeley Computer Aided Manufacturing Group, UC Berkeley, Dec. 1994.

[7.8] G. S. May and C. J. Spanos, "Automated Malfunction Diagnosis of Semiconductor Fabrication Equipment: A Plasma Etch Application," *IEEE Trans. Semiconductor Manufacturing*, vol. 6, no. 1, Feb. 1993, pp. 28-40.

[7.9] S. W. Butler, J. A. Stefani, "Supervisory Run-to-Run Control of Polysilicon Gate Etch Using In Situ Ellipsometry," *IEEE Trans. Semiconductor Manufacturing*, vol. 7, no. 2, May 1994, pp. 193-201.

[7.10] S. Maung, S. Banerjee, D. Draheim, S. Henck, and S. W. Butler, "Integration of In Situ Spectral Ellipsometry with MMST Machine Control," *IEEE Trans. Semiconductor Manufacturing*, vol. 7, no. 2, May 1994, pp. 184-192.

Appendix A

Test Structure Process Steps for Lam Rainbow 4400 Experiments

The following is the sequence of steps used to fabricate the test structures used in the Lam Rainbow 4400 Experiments.

1. p-type B<100> wafers, 14-22 ohms*cm.

2. Gate ox: Tylan5, 2.5 hours at 950°C. recipe: sgateox (~580A)

3. n+ doped poly: Tylan 11, 3.5 hours recipe: sdopolyh (~6000A)

   - Use the center boat, grow one lot of 12 wafers at a time.
   - Tylan7 anneal 15 min at 950°C.
   - Anneal all 24 wafers together.

4. LTO: Tylan 12, 450C O2:SiH4 = 90sccm: 60sccm 16min recipe: vdoltoc (~3000A)

- Use the rear boat, grow one lot of 12 wafers at a time.

5. Mask 1: Hardbake at least 40 min

   - HMDS 2 - 3 min.
   - Eaton I-line resist, standard process (#15), resist thickness ~0.9um
   - GCA expose mask 1 at standard focus, at exposure dose that can resolve 0.8um elbows. This is generally 31% more than the standard.
   - Post-exposure bake 60 sec. at 120°C
   - MTI develop standard recipe (#70)
   - Technics-C descum for 1 min. at 50W
   - Hardbake 20 min. at 120°C

6. Etch LTO: lam2 standard recipe, 750W, 85%endpoint, 30sec. overetch

7. Etch poly: lam4 standard recipe (Cl_2: He). Etch to endpoint.

8. Strip resist: Technics-C, 400W, 7min.

9. Mask 2: Hardbake at least 40 min.

   - HMDS 2 - 3 min.
   - Eaton I-line resist, standard process (#15), resist thickness ~1.1um
   - GCA expose mask 2 at standard focus, at exposure dose that ca resolve 0.8um elbows
   - Post-exposure bake 60 sec. at 120°C
   - MTI develop standard recipe (#70)
   - Technics-C descum for 1 min. at 50W
   - Hardbake 20 min. at 120°C

10. Etch LTO: lam2 standard recipe 850W 95% endpoint, 30sec. overetch

11. Strip resist: Technics-C, 400W, 7min.

12. Mask 3: Hardbake at least overnight.

    - HMDS 2 - 3 min.
    - Eaton I-line resist, standard process (#15), resist thickness ~1.1um
    - GCA expose mask 3 at standard focus, and at exposure dose that can resolve 0.8um elbows
    - Post-exposure bake 60 sec. at 120^{o}C
    - MTI develop standard recipe (#70)
    - Technics-c descum for 1 min. at 50W
    - Hardbake 30 min. at 120^{o}C

Appendix B

Discriminant Analysis Algorithm

The following C code generates the S-PLUS code used to train the discriminant analysis algorithm. Inputs to the code are: (1) the name of the data file, (2) the number of faults that you would like to diagnose, (3) the number of real-time signals used in the analysis, and (4) the name of the output file. The program will then prompt the user for the number of runs per fault. A sample output is included after the program.

```
#include <stdlib.h>
#include <stdio.h>
#include "math.h"

main(int argc, char *argv[])
{

 FILE *foutput;

 /* Set some defaults */
 char *fdata = NULL;
 int faultnum = -1;
 int parnum = -1;
 int i,k,temp,temp1;
 int flen[11];
 int total;

/*************************************************/
/*    This version checks the number of      */
/*    points per fault, so that you can      */
/*    have a different number of runs per    */
/*    fault.                         */
/*************************************************/

/******* Check that input is correct **********/
/* Input sequence is:
S_code.exe = executable program name
filename = datafilename
faultnum = number of faults
parnum = number of parameters
output = output file name
flen[i] = number of points per fault i
*/

  for (i = 1; i < argc; i++) {
if (strcmp("-p", argv[i]) == 0) {
   parnum = atoi(argv[++i]);
} else if (strcmp("-n", argv[i]) == 0) {
   faultnum = atoi(argv[++i]);
} else if (strcmp("-d",argv[i]) == 0) {
   fdata = argv[++i];
} else  {
     if ((foutput = fopen(argv[i],"a")) == NULL) {
    printf("Cannot open file %s\n",argv[i]);
    exit(1);
   }
}
   }

  /* check that all parameters were entered;
  if not, print out usage message only;  */
  if ((faultnum == -1) || (parnum == -1) ||
     (fdata == NULL)) {
print_usage();
exit(1);
  }

  /* Input number of points per fault */
  for (i = 1; i < faultnum + 1; i++) {
   printf("Enter the number of points for fault %d\n",i);
   scanf("%d",&flen[i]);
  }

/********* Start Writing Splus File *************/

/** Input **/
fprintf(foutput,"Data_matrix(scan(\"%s\"), ncol=%d,\
```

```
byrow=TRUE)\n\n",fdata, parnum);

/* Separate fault data, calculate means */
fprintf(foutput,"F1_Data[1:%d,]\n",flen[1]);
for (i = 2; i < faultnum + 1; i++) {
temp = 0;
temp1 = 0;
 for (k = 1; k < i; k++) {
  temp = temp + flen[k];
 }
 temp = temp + 1;
 for (k = 1; k < i+1; k++) {
  temp1 = temp1 + flen[k];
 }
 fprintf(foutput,"F%d_Data[%d:%d,]\n",i,temp,temp1);
}

for (i = 1; i < faultnum + 1; i++) {
 fprintf(foutput,"meanF%d_apply(F%d,2,mean)\n",i,i);
 fprintf(foutput,"covF%d_var(F%d)\n",i,i);
}

/* Linear discriminant function */
fprintf(foutput,"\n# Linear Discriminant Function\n");
fprintf(foutput,"k_c(");
for (i = 1; i < faultnum; i++) {
 fprintf(foutput,"%d,",flen[i]);
}
fprintf(foutput,"%d)\n",flen[faultnum]);

fprintf(foutput,"Fldf_Data%%*%%discr(Data,k)$vars\n\n");

fprintf(foutput,"# First ldf\n");

fprintf(foutput,"F1ldf1_Fldf[1:%d,1]\n",flen[1]);
for (i = 2; i < faultnum + 1; i++) {
temp = 0;
temp1 = 0;
 for (k = 1; k < i; k++) {
  temp = temp + flen[k];
 }
 temp = temp + 1;
 for (k = 1; k < i+1; k++) {
  temp1 = temp1 + flen[k];
 }
 fprintf(foutput,"F%dldf1_Fldf[%d:%d,1]\n",i,temp,temp1);
}

fprintf(foutput,"\n# Second ldf\n");
fprintf(foutput,"F1ldf2_Fldf[1:%d,2]\n",flen[1]);
for (i = 2; i < faultnum + 1; i++) {
temp = 0;
temp1 = 0;
 for (k = 1; k < i; k++) {
  temp = temp + flen[k];
 }
 temp = temp + 1;

 for (k = 1; k < i+1; k++) {
  temp1 = temp1 + flen[k];
 }

 fprintf(foutput,"F%dldf2_Fldf[%d:%d,2]\n",i,temp,temp1);
}

/* Matplots */
fprintf(foutput,"\n# Matplot, set up columns\n");
```

```
fprintf(foutput,"Fldf1_cbind\(");
for (i = 1; i < faultnum ; i++) {
fprintf(foutput,"F%dldf1,",i);
}
fprintf(foutput,"F%dldf1)\n",faultnum);

fprintf(foutput,"Fldf2_cbind\(");
for (i = 1; i < faultnum ; i++) {
fprintf(foutput,"F%dldf2,",i);
}
fprintf(foutput,"F%dldf2)\n\n",faultnum);
  /* Coefficients */
fprintf(foutput,"# Solve for the coefficients\n\n");
for (i = 1; i < faultnum ; i++) {
fprintf(foutput,"coeff%d_discr(Data,k)$vars[,1:%d]\n",i,i);
}
 /* Transform means */
fprintf(foutput,"# Find the transformed means of each fault\n");
fprintf(foutput,"# in two-dimensions\n\n");
for (i = 1; i < faultnum +1; i++) {
fprintf(foutput,"tmean%d_t(meanF%d)%%*%%coeff2 \n",i,i);
}


fprintf(foutput,"postscript(\"Matplot\")
matplot(Fldf1,Fldf2,col=1)
points(tmean1,pch=\"A\")
points(tmean2,pch=\"B\")
points(tmean3,pch=\"C\")
points(tmean4,pch=\"D\")\n");
fprintf(foutput,"\ngraphics.off()\n\n");

/* Classification */

/* Euclidean distance */
total = 0;
for  (i = 1; i < faultnum + 1; i++) {
total += flen[i];
}

fprintf(foutput,"# Euclidean distance\n\n");
fprintf(foutput,"EDloop_function(realfault,");
for (i = 1; i < faultnum+1; i++) {
 fprintf(foutput,"mean%d,",i);
}
fprintf(foutput,"coeff)\n");
fprintf(foutput,"\{\
   for (i in 1:%d)\n",total);
fprintf(foutput,"  {
F1_(realfault[i,])%%*%%coeff\n");
for (i = 1; i < faultnum + 1; i++) {
fprintf(foutput,"Mean%d_t(mean%d)%%*%%coeff\n",i,i);
}
for (i = 1; i < faultnum + 1; i++) {
fprintf(foutput,"dist%d_sqrt(sum( ((F1-Mean%d)^2) ))\n",i,i);
}
fprintf(foutput,"temp_min(");
for (i = 1; i < faultnum; i++) {
fprintf(foutput,"dist%d,",i);
}
fprintf(foutput,"dist%d)\n",faultnum);
for (i = 1; i < faultnum + 1; i++) {
fprintf(foutput,"if (temp == dist%d) j_%d\n",i,i);
}
fprintf(foutput,"if ((i<%d) && (j!=1)) print(paste(i,\" \
is misclassified as \",j))\n",flen[1]+1);
for (i = 2; i < faultnum + 1; i++) {
```

```
temp = 0;
temp1 = 0;
 for (k = 1; k < i; k++) {
  temp = temp + flen[k];
 }
 temp = temp + 1;
 for (k = 1; k < i+1; k++) {
  temp1 = temp1 + flen[k];
 }
fprintf(foutput,"if ((i>%d) && (i<%d) && (j!=%d))\n",
          temp,temp1,i);
fprintf(foutput,"  print(paste(i,\" is misclassified as \",j))\n");
}
fprintf(foutput,"  }\n\
}\n\n");
fprintf(foutput,"EDloop(Data,");
for (i = 1; i < faultnum+1; i++) {
 fprintf(foutput,"meanF%d,",i);
}
fprintf(foutput,"coeff1)\n");
/* Mahalanobis Distance */
fprintf(foutput,"\n# Mahalanobis distance \n\n");
fprintf(foutput,"Mahal_function(fault,");
for (i = 1; i < faultnum; i++) {
 fprintf(foutput,"mean%d,cov%d,",i,i);
}
fprintf(foutput,"mean%d,cov%d)\n",faultnum,faultnum);

fprintf(foutput,"{
for (i in 1:%d)\n",total);
fprintf(foutput,"  {");
for (i = 1; i < faultnum + 1; i++) {
    fprintf(foutput,"Mahal%d_t(fault[i,]-
mean%d)%%*%%%solve(cov%d)%%*%%%\
    (fault[i,]-mean%d)
    + log(det(cov%d))\n",i,i,i,i,i,i);
    }
    fprintf(foutput,"temp_min(");
    for (i = 1; i < faultnum; i++) {
     fprintf(foutput,"Mahal%d,",i);
    }
    fprintf(foutput,"Mahal%d)\n",faultnum);
    for (i = 1; i < faultnum + 1; i++) {
     fprintf(foutput,"if (temp == Mahal%d) j_%d\n",i,i);
    }

    fprintf(foutput,"if ((i<%d) && (j!=1)) print(paste(i,\" \
    is misclassified as \",j))\n",flen[1]+1);
    for (i = 2; i < faultnum + 1; i++) {
    temp = 0;
    temp1 = 0;
     for (k = 1; k < i; k++) {
      temp = temp + flen[k];
     }
     temp = temp + 1;
     for (k = 1; k < i+1; k++) {
      temp1 = temp1 + flen[k];
     }
    fprintf(foutput,"if ((i>%d) && (i<%d) && (j!=%d))\n",
              temp,temp1,i);
    fprintf(foutput,"  print(paste(i,\" is misclassified as \",j))\n");
    }
    fprintf(foutput,"  }
    }\n\n");
```

```
fprintf(foutput,"Mahal(Data,");
for (i = 1; i < faultnum + 1; i++) {
  fprintf(foutput,"meanF%d,covF%d,",i,i);
}
fprintf(foutput,"meanF%d,covF%d)\n",faultnum,faultnum);

} /* end main */

print_usage()
{
    printf("usage:\nS_code.exe -d filename -n faultnum -p parnum \
output\n");
}
```

The following S-PLUS output file is for 7 different faults, with a different number of runs per fault. (11 points for the first fault, 9 for the second, and 15 for each of the remaining five faults.) Six real-time signals are used for the training.

```
Data_matrix(scan("phs2_train.res"), ncol=5,byrow=TRUE)
NewData_matrix(scan("predR_diag.res"), ncol=5,byrow=TRUE)

F1_Data[1:11,]
F2_Data[12:20,]
F3_Data[21:35,]
F4_Data[36:50,]
F5_Data[51:65,]
F6_Data[66:80,]
F7_Data[81:95,]
meanF1_apply(F1,2,mean)
covF1_var(F1)
meanF2_apply(F2,2,mean)
covF2_var(F2)
meanF3_apply(F3,2,mean)
covF3_var(F3)
meanF4_apply(F4,2,mean)
covF4_var(F4)
meanF5_apply(F5,2,mean)
covF5_var(F5)
meanF6_apply(F6,2,mean)
covF6_var(F6)
meanF7_apply(F7,2,mean)
covF7_var(F7)
```

```
# Linear Discriminant Function
k_c(11,9,15,15,15,15,15)
Fldf_Data%*%discr(Data,k)$vars

# First ldf
F1ldf1_Fldf[1:11,1]
F2ldf1_Fldf[12:20,1]
F3ldf1_Fldf[21:35,1]
F4ldf1_Fldf[36:50,1]
F5ldf1_Fldf[51:65,1]
F6ldf1_Fldf[66:80,1]
F7ldf1_Fldf[81:95,1]

# Second ldf
F1ldf2_Fldf[1:11,2]
F2ldf2_Fldf[12:20,2]
F3ldf2_Fldf[21:35,2]
F4ldf2_Fldf[36:50,2]
F5ldf2_Fldf[51:65,2]
F6ldf2_Fldf[66:80,2]
F7ldf2_Fldf[81:95,2]

# First ldf for matplot
F1ldf1M_F1ldf1[1:9]
F2ldf1M_F2ldf1[1:9]
F3ldf1M_F3ldf1[1:9]
F4ldf1M_F4ldf1[1:9]
F5ldf1M_F5ldf1[1:9]
F6ldf1M_F6ldf1[1:9]
F7ldf1M_F7ldf1[1:9]

# Second ldf for matplot
F1ldf2M_F1ldf2[1:9]
F2ldf2M_F2ldf2[1:9]
F3ldf2M_F3ldf2[1:9]
F4ldf2M_F4ldf2[1:9]
F5ldf2M_F5ldf2[1:9]
F6ldf2M_F6ldf2[1:9]
F7ldf2M_F7ldf2[1:9]

# Matplot, set up columns
Fldf1_cbind(F1ldf1M,F2ldf1M,F3ldf1M,F4ldf1M,F5ldf1M,F6ldf1M,F7ldf1
M)
Fldf2_cbind(F1ldf2M,F2ldf2M,F3ldf2M,F4ldf2M,F5ldf2M,F6ldf2M,F7ldf2
M)

## The coeff. should be limited by the number of parameters.
# Solve for the coefficients

coeff1_discr(Data,k)$vars[,1:1]
coeff2_discr(Data,k)$vars[,1:2]
coeff3_discr(Data,k)$vars[,1:3]
coeff4_discr(Data,k)$vars[,1:4]
coeff5_discr(Data,k)$vars[,1:5]
coeff6_discr(Data,k)$vars[,1:6]
# Find the transformed means of each fault
# in two-dimensions

tmean1_t(meanF1)%*%coeff2
tmean2_t(meanF2)%*%coeff2
tmean3_t(meanF3)%*%coeff2
tmean4_t(meanF4)%*%coeff2
tmean5_t(meanF5)%*%coeff2
tmean6_t(meanF6)%*%coeff2
tmean7_t(meanF7)%*%coeff2
newpoints_NewData%*%coeff2
```

```
newpoint1_newpoints[3:11,1]
newpoint2_newpoints[3:11,2]

#postscript("Matplot")
matplot(Fldf1,Fldf2,col=1)
points(newpoints[1:2,],pch=0,cex=2)
points(newpoints[3:15,],pch=1,cex=2)

#points(tmean1,pch="A")
#points(tmean2,pch="B")
#points(tmean3,pch="C")
#points(tmean4,pch="D")

#graphics.off()

# Euclidean distance

EDloop_function(real-
fault,mean1,mean2,mean3,mean4,mean5,mean6,mean7,coeff)
{   for (i in 1:95)
  {
F1_(realfault[i,])%*%coeff
Mean1_t(mean1)%*%coeff
Mean2_t(mean2)%*%coeff
Mean3_t(mean3)%*%coeff
Mean4_t(mean4)%*%coeff
Mean5_t(mean5)%*%coeff
Mean6_t(mean6)%*%coeff
Mean7_t(mean7)%*%coeff
dist1_sqrt(sum( ((F1-Mean1)^2) ))
dist2_sqrt(sum( ((F1-Mean2)^2) ))
dist3_sqrt(sum( ((F1-Mean3)^2) ))
dist4_sqrt(sum( ((F1-Mean4)^2) ))
dist5_sqrt(sum( ((F1-Mean5)^2) ))
dist6_sqrt(sum( ((F1-Mean6)^2) ))
dist7_sqrt(sum( ((F1-Mean7)^2) ))
temp_min(dist1,dist2,dist3,dist4,dist5,dist6,dist7)
if (temp == dist1) j_1
if (temp == dist2) j_2
if (temp == dist3) j_3
if (temp == dist4) j_4
if (temp == dist5) j_5
if (temp == dist6) j_6
if (temp == dist7) j_7
if ((i<12) && (j!=1)) print(paste(i," is misclassified as ",j))
if ((i>12) && (i<20) && (j!=2))
  print(paste(i," is misclassified as ",j))
if ((i>21) && (i<35) && (j!=3))
  print(paste(i," is misclassified as ",j))
if ((i>36) && (i<50) && (j!=4))
  print(paste(i," is misclassified as ",j))
if ((i>51) && (i<65) && (j!=5))
  print(paste(i," is misclassified as ",j))
if ((i>66) && (i<80) && (j!=6))
  print(paste(i," is misclassified as ",j))
if ((i>81) && (i<95) && (j!=7))
  print(paste(i," is misclassified as ",j))
  }
}

EDloop(Data,meanF1,meanF2,meanF3,meanF4,meanF5,meanF6,meanF7,coe
ff1)

EDloop.class(New-
Data,meanF1,meanF2,meanF3,meanF4,meanF5,meanF6,meanF7,coeff2)
# Mahalanobis distance
```

```
    Mahal_function(fault,mean1,cov1,mean2,cov2,mean3,cov3,mean4,c
ov4,mean5,cov5,mean6,cov6,mean7,cov7)
    {
    for (i in 1:95)
     {Mahal1_t(fault[i,]-mean1)%*%solve(cov1)%*%(fault[i,]-mean1)
    + log(det(cov1))
    Mahal2_t(fault[i,]-mean2)%*%solve(cov2)%*%(fault[i,]-mean2)
    + log(det(cov2))
    Mahal3_t(fault[i,]-mean3)%*%solve(cov3)%*%(fault[i,]-mean3)
    + log(det(cov3))
    Mahal4_t(fault[i,]-mean4)%*%solve(cov4)%*%(fault[i,]-mean4)
    + log(det(cov4))
    Mahal5_t(fault[i,]-mean5)%*%solve(cov5)%*%(fault[i,]-mean5)
    + log(det(cov5))
    Mahal6_t(fault[i,]-mean6)%*%solve(cov6)%*%(fault[i,]-mean6)
    + log(det(cov6))
    Mahal7_t(fault[i,]-mean7)%*%solve(cov7)%*%(fault[i,]-mean7)
    + log(det(cov7))

temp_min(Mahal1,Mahal2,Mahal3,Mahal4,Mahal5,Mahal6,Mahal7)
    if (temp == Mahal1) j_1
    if (temp == Mahal2) j_2
    if (temp == Mahal3) j_3
    if (temp == Mahal4) j_4
    if (temp == Mahal5) j_5
    if (temp == Mahal6) j_6
    if (temp == Mahal7) j_7
    if ((i<12) && (j!=1)) print(paste(i," is misclassified as ",j))
    if ((i>12) && (i<20) && (j!=2))
     print(paste(i," is misclassified as ",j))
    if ((i>21) && (i<35) && (j!=3))
     print(paste(i," is misclassified as ",j))
    if ((i>36) && (i<50) && (j!=4))
     print(paste(i," is misclassified as ",j))
    if ((i>51) && (i<65) && (j!=5))
     print(paste(i," is misclassified as ",j))
    if ((i>66) && (i<80) && (j!=6))
     print(paste(i," is misclassified as ",j))
    if ((i>81) && (i<95) && (j!=7))
     print(paste(i," is misclassified as ",j))
     }
    }

    Mahal(Data,meanF1,covF1,meanF2,covF2,meanF3,covF3,meanF4,covF4,mea
nF5,covF5,meanF6,covF6,meanF7,covF7,meanF7,covF7)
```

Appendix C

Staged Clustering and Neural Networks Algorithm

The following S-PLUS code performs the staged clustering section of the algorithm. The code shown on the following pages is specific for the Lam Rainbow 4400 polysilicon etcher.

```
##############################################
## The complete algorithm to separate faults
## from Training PhaseII and Verification Experiments
##############################################

##Run from ~sflee
##collection("MDS")
##X11()

##rt.both.means_cbind(rpm.m,rt.means)
# eliminate wafers with unstable signals
##rt_rt.both.means[-c(4,5,15,17,23,24,25,26,31,32),]
##pred_cbind(pred.rpm.m,pred.rt.means)

stnd_function(mat,sdev,cnt)
{
n_nrow(mat);p_ncol(mat)
mat2_matrix(0,n,p)
 for (i in (1:n)) {
  mat2[i,]_(mat[i,] - cnt)/sdev
  }
  mat2
}

filt_function(mat)
{
n_nrow(mat);p_ncol(mat)
for (i in 1:n){
  for (j in 1:p){
    if ( abs(mat[i,j]) <= 3.0) mat[i,j]_0
    }
  }
  mat
}

#### STEP 1:
# Subtract off the center points, cnt1 for the central composite
# and cnt2 for the verification experiment.

varcnt1_apply(cntpts[1:2,],2,var);varcnt2_apply(cntpts[3:6,],2,var)
sdevcnt1_sqrt(varcnt1);sdevcnt2_sqrt(varcnt2)

rt2.stnd_stnd(rt,sdevcnt1,cnt1)
pred2.stnd_stnd(pred,sdevcnt2,cnt2)

## Use the t-test to post-filter.
rt2.filt_filt(rt2.stnd)
pred2.filt_filt(pred2.stnd)

#### STEP 2:
## Use the trends of pred2 and rt2 to separate
## the increases and decreases, and to determine
## which signs should be flipped.

both.filt_rbind(rt2.filt[19:27,],pred2.filt[6,])
both.sign_sign(both.filt)
#head_c("Gd", "Rd", "C5","Tu","Wd","C6","Ru","Td","Pu",
#"unknown")

## Remove Coil, DCB (duplicate), Imp(duplicate), Tune, Volt
both2.sign_both.sign[,-c(7,8,10,12,13)]
both2.dist_dist(both2.sign,metric="euclidean")
both2.hclust_hclust(both2.dist,method="com")
#plclust(both2.hclust,label=head)
#title("STEP 2: Use the trends to determine which signs
#should be flipped")
```

```
step2.cl_cutree(both2.hclust,h=4.0)

#### STEP 3:
## Split data into 2 clusters, negate all signs
## from one cluster, then recluster, separating
## the centerpoints and power changes from the rest.

## Change signs
both3.sign_both2.sign

## Determine how many runs are in each cluster
n_nrow(both3.sign)
l_0
for (i in 1:n) {
if (step2.cl[i] == 2) l_l+1
}

both3.sign[both2.hclust$order[(l+1):n],]_
  (-1)*both2.sign[both2.hclust$order[(l+1):n],]

## Change signs of original "rtpred" matrix
neg.filt_both.filt
neg.filt[both2.hclust$order[(l+1):n],]_
  (-1)*rtpred[both2.hclust$order[(l+1):n],]

both3.dist_dist(both3.sign[,1:2],metric="euclidean")
both3.hclust_hclust(both3.dist,method="com")
#plclust(both3.hclust,label=head)
#title("STEP 3: Separate the Centerpoints and Power
#Changes from the rest")

## Separate Centerpoints and Power
cent_0; power_0
if (dist(both3.sign[c(3,10),],metric="euclidean") < 0.25) cent_1
if (dist(both3.sign[c(6,10),],metric="euclidean") < 0.25) cent_1
if (dist(both3.sign[c(5,10),],metric="euclidean") < 0.5) power_1

## Print out diagnosed power faults
if (power == 1) print("The wafer has a power problem")

## Print out wafers with no problems
if (cent == 1) print("The wafer is classified as a centerpoint")

if ( (cent != 1) && (power != 1) ) {
#### STEP 4:
## Examine DCB to filter out the Gap

tdata_neg.filt[-c(3,5,6),]
head2_head[-c(3,5,6)]
PRGT.dist_dist(tdata[,6])
PRGT.hclust_hclust(PRGT.dist,method="com")
#plclust(PRGT.hclust,label=head2)
#title("STEP 4: Filter out Gap")

gap_0
if (neg.filt[10,6] > 3.0) gap_1

## Print out diagnosed gap faults
if (gap == 1) print("The wafer has a gap problem")


if (gap != 1) {
## Retain undiagnosed faults
PRT.data_neg.filt[-c(1,3,5,6),]
```

```
#### STEP 5:
## Examine VRF, IRF, Z, angle, DCB, Enda, and Phase to
## separate Total Flow from Pressure and Ratio

head3_head2[-c(1)]
PRT_PRT.data[,c(2,5,11)]
PRT.dist_dist(PRT,metric="euclidean")
PRT.hclust_hclust(PRT.dist,method="com")
#plclust(PRT.hclust,label=head3)
#title("STEP 5: Filter out Total Flow")

total_0
if (dist(neg.filt[c(4,10),],metric="euclidean") < 30) total_1
if (dist(neg.filt[c(8,10),],metric="euclidean") < 30) total_1

## Print out diagnosed total flow faults
if (total == 1) print("The wafer has a total flow problem")

if (total != 1) {
## Retain undiagnosed faults
PR.data_neg.filt[-c(1,3,5,6,4,8),]

#### STEP 6:
## Separate as best as possible Pressure and Ratio

head4_head3[-c(2,4)]

PR_PR.data[,c(2,3,4,5)]
PR.dist_dist(PR,metric="euclidean")
PR.hclust_hclust(PR.dist,method="com")
#plclust(PR.hclust,label=head4)
#title("STEP 6: Best division of Ratio and Pressure")

press_0; ratio_0
if (dist(neg.filt[c(9,10),c(2,3,4,5)],metric="euclidean") < 30) press_1
if (dist(neg.filt[c(2,10),c(2,3,4,5)],metric="euclidean") < 30) ratio_1
if (dist(neg.filt[c(7,10),c(2,3,4,5)],metric="euclidean") < 30) ratio_1

## Print out diagnosed pressure and ratio faults
if (press == 1) print("The wafer has a pressure problem")
if (ratio == 1) print("The wafer has a ratio of flows problem")


} # total
} # gap
} # cent, power
```