IEEE TRANSACTIONS ON SEMICONDUCTOR MANUFACTURING, VOL. 8, NO. 3, AUGUST 1995

Prediction of Wafer State After Plasma Processing Using Real-Time Tool Data

Sherry F. Lee, *Member, IEEE*, and Costas J. Spanos, *Member, IEEE*

Abstract— **Empirical models based on real-time equipment signals are used to predict the outcome (e.g., etch rates and uniformity) of each wafer during and after plasma processing. Three regression and one neural network modeling methods were investigated. The models are verified on data collected several weeks after the initial experiment, demonstrating that the models built with real-time data survive small changes in the machine due to normal operation and maintenance. The predictive capability can be used to assess the quality of the wafers after processing, thereby ensuring that only wafers worth processing continue down the fabrication line. Future applications include real-time evaluation of wafer features and economical run-to-run control.**

I. Introduction

With increasing world-wide competition and escalating factory costs, companies are continuously improving their manufacturing skills to maintain high yield, increase throughput, and reduce the cost of equipment ownership on the manufacturing line. A key element in achieving these goals is to monitor the equipment to ensure that the semiconductor wafers are processed properly at each step. The cost in dollars and throughput of measuring each wafer after it completes each step, however, becomes prohibitive in semiconductor factories producing hundreds of 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. 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.

We propose to use empirical models based on real-time equipment data to predict the outcome of each wafer immediately after processing by each piece of equipment [1]. This will reduce the need for costly and time-consuming wafer measurements. The prediction ability allows the quality of the wafer to be known immediately after processing, thereby obtaining important wafer yield information to ensure that only wafers worth processing continue down the line. By predicting the wafer characteristics, significant cost reduction is possible, thus lowering the overall cost of equipment ownership [2].

We verify this general prediction methodology on a plasma etcher, one of the costliest pieces of equipment in the semiconductor fabrication line. Not only is the etcher usually a bottleneck piece of equipment, the scrap produced by the etcher can be extremely costly. Furthermore, empirical models are appropriate because the etching mechanisms are not well understood. Although there is a tremendous push to develop models relating the plasma to interesting output characteristics of the wafer based on basic physical principles, first principle models are several years away from becoming useful on the factory floor [3]–[5]. Thus, at this time empirical models are faster and more practical for wafer state prediction.

To provide useful prediction capabilities, robust prediction models of the plasma etchers are required. The industry standard is to use response surface methodology (RSM) to build models relating the input settings of the etchers to the output wafer state (Fig. 1). Models using input settings, however, may 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. Wangmaneerat [6] used partial least squares regression to model the etch rate of silicon nitride thin films systems with optical emission spectroscopy (OES) signals. More recently, Anderson *et al.* [7] demonstrated that spatially resolved OES signals are effective in modeling plasma etch rates, selectivities, and uniformity, also using partial least squares regression. Neither work, however, has shown prediction capabilities by testing the models on data not used to build the original models. Rietman and Lory [8] have shown that neural networks can be used to model wafer attributes using a combination of real-time tool data and input setting data. The output of the model was the final oxide thickness in the source and drain regions of CMOS devices. The inputs to the model included input settings such as applied RF power, chamber pressure, gas flow rates, and real-time data such as induced dc bias, reflected RF power, and the emission spectrum, as well as the etch time. The resulting neural network models were tested using data not used to build the model. This testing data, however, was not separated in time from the original experiment, so it did not necessarily test the model's ability to withstand normal equipment drifts due to use over time.

This paper shows that successful wafer state prediction over long periods of time can be achieved by using the real-time data from key sensors inside the equipment. Because

Manuscript received August 24, 1994; revised March 31, 1995. This work was supported by the SRC under Grant 93-MP-700, 94-MY-700, Digital Equipment Corporation, and Lam Research.

S. Lee was with the Department of Electrical Engineering & Computer Sciences, University of California at Berkeley, CA 94720 USA. She is now with Motorola, Irvine, CA 92718 USA.

C. Spanos is with the Department of Electrical Engineering & Computer Sciences, University of California, Berkeley, CA 94720 USA.

IEEE Log Number 9412874.

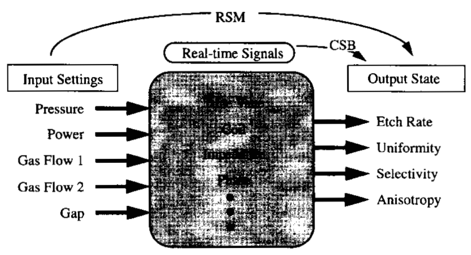


Fig. 1. Wafer state prediction: This paper shows that Chamber State Based (CSB) models, which map the chamber state data to the output state, are effective for prediction of wafer state, even in the presence of equipment aging.

these real-time signals provide important information about the chamber state, we call the signals *chamber state* data. Models built with chamber state data, called CSB models, are effective for prediction since the chamber state data reflects the actual (as opposed to the intended) state of the equipment.

To develop the prediction models, two sets of experiments were conducted. During the experiments, both the input settings and the chamber state data were collected. The wafer states of interest are the etch rates, selectivity, and uniformity. The first experiment, called the *training experiment,* consists of a central composite design. The models using data from the training experiment relating the chamber state data to the wafer states are called the training models. The second experiment, called the *verification experiment,* was conducted several weeks later to determine the actual prediction capability of the training models. Three types of regression modeling methods for prediction (ordinary least squares regression, principal component regression, and partial least squares regression) are explored. These regression models are also compared to models developed using simple neural networks. Neural networks are included in this study because they have emerged as an effective modeling method for semiconductor manufacturing processes. In addition, it has been shown that neural networks result in superior prediction results compared to ordinary least squares regression using input settings [8]–[12]. In this paper, we compare different regression techniques with a simple feed-forward neural network using real-time data. The prediction metric used to compare the models 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 which were not included in model generation.

The goal of this paper, then, is to show that chamber state 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 helps determine the prediction capability of the models. The paper begins with a description of the chamber state signals used in the CSB models, followed by a discussion of the methodology and models used to determine the wafer state prediction capability of the models. Next is a description of the training and verification experiments. The modeling results are then discussed, followed by a brief discussion of future directions.

TABLE I
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 II
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 $90°$) 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

II. CHAMBER STATE DATA

The chamber state data collected from the plasma etcher consist of various electrical and mechanical signals. Generally, over 200 signals can be collected from modern plasma etchers [13]. In this work, between six and thirteen of these signals are used. Six signals are collected or calculated via a Comdel Real Power Monitor (RPM-1), which sits after the matching network and directly above the upper electrode [14]. The signals are read through its own RS232 interface. The other seven signals are collected via the Brookside LamStation software package, which reads the signals from the SECS-II (SEMI Equipment Communication Standard-II) serial port on the etcher [15]. The signals monitored by each data collection system are listed in Table I. A brief description of each signal is listed in Table II. 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 statistically correlated, these signals are not identical.

Fig. 2 shows the real-time signals of the RF load coil position and dc bias for different fixed input conditions on each of 12 wafers. Approximately 30 points are collected per signal, per wafer etched. Since the data are collected 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. The time series signatures of the signals are exploited to determine outliers. For example, notice the instability in wafers #4 and #5 shown in Fig. 2. For an unknown reason, the RF power dropped significantly during the processing of wafer #4, causing corresponding adjust ments in both coil position and dc bias. It turns out that the etch rate

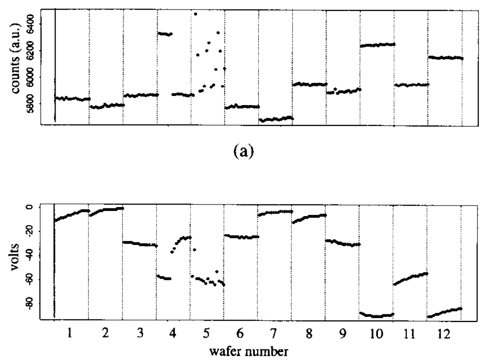


Fig. 2. Real-time chamber state signals of (a) RF load coil position and (b) dc bias for different input conditions on 12 wafers. Wafers #4 and #5 have unstable real-time signals and are rejected as "bad" wafers [17]. Notice the large wafer-to-wafer variance compared to the within-wafer variance.

for wafer #4 was unusually low due to the drop in RF power. Therefore, the run corresponding to wafer #4 was left out of the model training runs. As seen in Fig. 2, wafer #5 exhibited unstable signals and was also rejected as an outlier.[1]

Once the outliers have been determined, in this work the time series nature of the signals is not used for prediction purposes. Instead, the wafer-to-wafer variability is mapped to the output wafer state. Fig. 2 shows (excluding wafers #4 and #5) that the wafer-to-wafer variance is much larger than the within-wafer variance. Therefore the average values per signal across each wafer can be used as the input for the prediction models built with the real-time signals. Each signal is averaged over the duration of the main etch step (after the native oxide breakthrough etch and before the overetch), which lasts approximately 30 s.

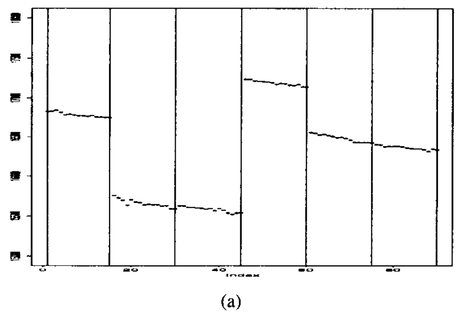
Unlike the fixed input settings, the chamber state signals change with the state of the machine. This is illustrated in Fig. 3, which shows the load impedance and RF tune vane position for the duration of six wafers processed at the same input settings. While the input settings are fixed for all six wafers, the chamber state signals vary for each etch, indicating that the chamber state data may give a more accurate description of the actual equipment state.

Although the examples shown in this paper are based on data collected from the LamStation and RPM-1 sensors, the methodology presented is general and can be applied to other types of 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 describe the chamber state. At present we have found the data collected from LamStation and RPM-1 to be sufficient to show the power of this class of real-time tool data.

III. Wafer State Prediction Methodology

This section outlines the basic advantages and disadvantages of the four modeling methods, and discusses the prediction metric used to compare the prediction capability of the models.

[1]Although undetected by the machine, these errors were detected by the Berkeley real-time fault detection system, RTSPC [16], [17].



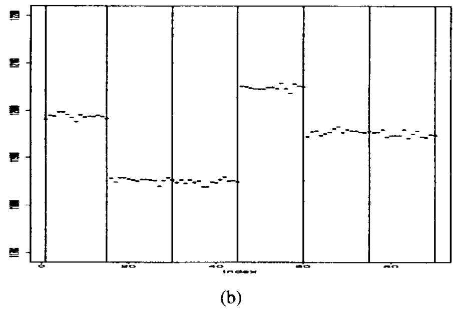


Fig. 3. Real-time signals for six center point wafers during the duration of the main etch. Unlike the input settings, the real-time chamber state signals (a) load impedance and (b) RF tune vane position reflect changes in machine state.

A. Modeling Methods

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 error back propagation neural networks are also briefly discussed.

1) Ordinary Least Squares Regression: The first regression method discussed is ordinary least squares regression (OLSR). The equation for the linear regression model is[2]

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\underset{\sim}{\beta}} \tag{1}$$

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

$$\hat{\underset{\sim}{\beta}} = (\boldsymbol{X}'\boldsymbol{X})^{-1}\boldsymbol{X}'\boldsymbol{y} \tag{2}$$

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

Prediction problems arise when the columns of $\boldsymbol{X}$ exhibit multicollinearity, or are highly correlated. The main idea is that high correlation in $\boldsymbol{X}$ leads to small eigenvalues in $\boldsymbol{X}'\boldsymbol{X}$,

[2]In this paper, 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 ($'$).

which results in a high variance in both the estimate of the coefficients and the predicted responses. For example, let $\hat{y}_o = \boldsymbol{x}_o\hat{\underset{\sim}{\beta}}$ be a predicted value. The variance of this predicted value can be solved in terms of the eigenvalues w_j and eigenvectors $\boldsymbol{v}_j$ of $\boldsymbol{X}'\boldsymbol{X}$:

$$\begin{aligned}\mathrm{var}\,(\hat{y}_o) &= \mathrm{var}\,(\boldsymbol{x}_o\hat{\underset{\sim}{\beta}}) = \boldsymbol{x}_o\mathrm{cov}\,[\hat{\underset{\sim}{\beta}},\hat{\underset{\sim}{\beta}}]\boldsymbol{x}_o' \\ &= \sigma^2\boldsymbol{x}_o\sum_{j=1}^{p}\frac{\boldsymbol{v}_j\boldsymbol{v}_j'}{w_j}\boldsymbol{x}_o' = \sigma^2\sum_{j=1}^{p}\frac{\boldsymbol{x}_o\boldsymbol{v}_j\boldsymbol{v}_j'\boldsymbol{x}_o'}{w_j} \qquad (3)\end{aligned}$$

where $\mathrm{cov}[\boldsymbol{Y},\boldsymbol{Y}] = \sigma^2\boldsymbol{I}_n$. Equation (3) shows that the variance of the predicted values depends on both the value of the eigenvalues and the direction of the input $\boldsymbol{x}_o$. The variance will be large for small eigenvalues and large values of $\boldsymbol{x}_o\boldsymbol{v}_j$. 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 $\boldsymbol{X}$ exhibit multicollinearity, the prediction capability of the model can be very poor.

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 parameters $\boldsymbol{X}$. This number can easily escalate to an almost unmanageable number when interactions are included. For example, in this paper 13 main signals are collected, resulting in 90 model variables when all the corresponding two-way interactions are included. Because many of the signals are correlated, not all 90 variables should (or can) be used independently in a model.

To eliminate the correlation among the input variables, PCR transforms the correlated input variables to a set of orthogonal variables. The transformed variables $\boldsymbol{Z}$, 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 $\boldsymbol{V}$ of $\boldsymbol{X}'\boldsymbol{X}$. The equation for the transformed variables $\boldsymbol{Z}$ is

$$\boldsymbol{Z} = (\boldsymbol{X} - \boldsymbol{1}\overline{\boldsymbol{x}}')\boldsymbol{V} \qquad (4)$$

where $\overline{\boldsymbol{x}}'$ is the vector of average values of each variable in $\boldsymbol{X}$ and $\boldsymbol{1}$ 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

$$\hat{\boldsymbol{y}} = \boldsymbol{Z}\hat{\underset{\sim}{\gamma}} \qquad (5)$$

where $\hat{\underset{\sim}{\gamma}}$ is the estimate of the coefficients using the equation

$$\hat{\underset{\sim}{\gamma}} = (\boldsymbol{Z}'\boldsymbol{Z})^{-1}\boldsymbol{Z}'\boldsymbol{y}.$$

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. The subset of statistically significant PC's in the model are determined by calculating the *Student-t test* for each of the coefficients. Only those PC's with statistically significant coefficients at a specified level are retained in the model (0.05 significance level is used in the examples of Section V).

While PCR decreases the number of parameters in the model, each model parameter still consists of a linear combination of input variables. Ideally, those input variables in $\boldsymbol{X}$ 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 $\boldsymbol{X}$. Assuming that the model is of the form in (5) and using (4) to substitute in for $\boldsymbol{Z}$

$$\hat{\boldsymbol{y}} = (\boldsymbol{X} - \boldsymbol{1}\overline{\boldsymbol{x}}')\boldsymbol{V}\hat{\underset{\sim}{\gamma}} = \boldsymbol{X}\boldsymbol{V}\hat{\underset{\sim}{\gamma}} - \boldsymbol{1}\overline{\boldsymbol{x}}\boldsymbol{V}\hat{\underset{\sim}{\gamma}} = \boldsymbol{X}\hat{\underset{\sim}{\beta}} - \boldsymbol{1}\overline{\boldsymbol{x}}\hat{\underset{\sim}{\beta}} \qquad (6)$$

where $\hat{\underset{\sim}{\beta}} = \boldsymbol{V}\hat{\underset{\sim}{\gamma}}$. The general rule of thumb we found was to eliminate those input parameters which have $\hat{\underset{\sim}{\beta}}$ values at least a magnitude smaller than the average of the largest $\hat{\underset{\sim}{\beta}}$ values. Regenerate the PCR model with the reduced set of input parameters, using the *Student-t* test to calculate the significance of the new PC's. Continue to reduce the input parameter space as described above until the model prediction no longer improves. (An effective metric to determine prediction is described in Section III.B) This simple, yet effective empirical method handles large numbers of input parameters very easily.

3) Partial Least Squares Regression: The last regression 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 [18]. 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 $\boldsymbol{Y}$. 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 $a = 1, 2, \cdots, A_{\max}$ [18]–[20]:

1) Determine the loading weight vector $\hat{\boldsymbol{w}}_a$:

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

The loadings $\hat{\boldsymbol{w}}_a$ are orthonormal vectors which maximize the covariance between $\boldsymbol{X}_{a-1}$ and $\boldsymbol{y}_{a-1}$. In other words, $\hat{\boldsymbol{W}} = (\hat{\boldsymbol{w}}_1, \hat{\boldsymbol{w}}_2, \cdots, \hat{\boldsymbol{w}}_A)$ relates the input and response, and is used to calculate the response in the model.

2) Estimate the scores $\hat{\boldsymbol{t}}_a$:

$$\hat{\boldsymbol{t}}_a = \boldsymbol{X}_{a-1}\hat{\boldsymbol{w}}_a.$$

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

3) Estimate the input loadings $\hat{\boldsymbol{p}}_a$:

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

$\hat{\boldsymbol{p}} = (\hat{\boldsymbol{p}}_1, \hat{\boldsymbol{p}}_2, \cdots, \hat{\boldsymbol{p}}_A)$ is similar to the eigenvector matrix $\boldsymbol{V}$ in PCR, in that it consists of the loadings for the input. Although $\hat{\boldsymbol{p}}$ is chosen to ensure that the $\hat{\boldsymbol{t}}_a$ vectors are orthogonal, the $\hat{\boldsymbol{p}}_a$ vectors are generally not orthogonal. Unlike the loadings in PCA, the first $\hat{\boldsymbol{p}}_a$ 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 $\hat{q}_a$:

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

$\hat{\boldsymbol{Q}} = (\hat{q}_1, \hat{q}_2, \cdots, \hat{q}_A)$ is the additional loading term which brings the response into the model. It relates the score $\hat{\boldsymbol{t}}_a$ 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 $\hat{\boldsymbol{F}}$ by subtracting the estimated values found in the previous steps from the actual values:

$$\hat{\varepsilon} = \boldsymbol{X}_{a-1} - \hat{\boldsymbol{t}}_a\hat{\boldsymbol{p}}'_a$$
$$\hat{\boldsymbol{F}} = \boldsymbol{y}_{a-1} - \hat{\boldsymbol{t}}_a\hat{q}_a.$$

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

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

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 $\hat{\boldsymbol{y}} = \boldsymbol{1}\hat{\beta}_o + \boldsymbol{X}\underset{\sim}{\hat{\beta}}$ are

$$\underset{\sim}{\hat{\beta}} = \hat{\boldsymbol{W}}(\hat{\boldsymbol{P}}'\hat{\boldsymbol{W}})^{-1}\hat{\boldsymbol{q}} \quad \text{and} \quad \hat{\beta}_o = \bar{y} - \bar{\boldsymbol{x}}'\underset{\sim}{\hat{\beta}}. \tag{7}$$

Using (7) 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.

4) Feed-Forward Error Backward Propagation Neural Networks: The last modeling method investigated is neural networks, which 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 [8], [9], [11].

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. In this application, one hidden layer was used, making a total of three layers in the network. 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 $f(x) = 1/1 + e^{-x}$, where x is the sum of the weighted outputs of the nodes preceding this particular node.

The neural network algorithm selected for this analysis is the feed-forward, error backward propagation (FFEBP) method, which has shown to be effective in modelling noisy input and out put data [9]–[11]. In this algorithm, the inputs are fed forward through the layers of the net work until reaching the output layer. The result at the output layer of node j is compared with the desired, or training, output. The difference, called the error, is used with the output of node i in a neighboring layer 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 [21]. In the FFEBP method, the gradient search method is used to minimize the sum of the squared errors [22]. A more thorough description of the algorithm can be found the review paper by Widrow and Lehr [23].

The Stuttgart Neural Network Simulator (SNNS) was used to simulate and train the neural networks [21]. 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.

B. 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 I, two sets of experiments were conducted—the first for model generation and the second for model verification. 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

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase® Smarter legal research.