A QUALITATIVE MODELING FRAMEWORK FOR SEMICONDUCTOR MANUFACTURING PROCESSES

—Self-Learning Fuzzy Inference System and the Statistical Analysis of Categorical Data

by

Raymond L. Chen

Memorandum No. UCB/ERL M95/6

25 January 1995

A QUALITATIVE MODELING FRAMEWORK FOR SEMICONDUCTOR MANUFACTURING PROCESSES

—Self-Learning Fuzzy Inference System and the Statistical Analysis of Categorical Data

by

Raymond L. Chen

Memorandum No. UCB/ERL M95/6

25 January 1995

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A QUALITATIVE MODELING FRAMEWORK FOR SEMICONDUCTOR MANUFACTURING PROCESSES

—Self-Learning Fuzzy Inference System and the Statistical Analysis of Categorical Data

by

Raymond L. Chen

Memorandum No. UCB/ERL M95/6

25 January 1995

ELECTRONICS RESEARCH LABORATORY

College of Engineering
University of California, Berkeley
94720

A Qualitative Modeling Framework for Semiconductor Manufacturing Processes

— Self-Learning Fuzzy Inference System and the Statistical Analysis of Categorical Data

Raymond L. Chen

Dissertation, Doctor of Philosophy in Engineering
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley

Signature: _____
Committee Chair

Abstract

Many qualitative properties of the product and the process are of interest during semiconductor manufacturing. Typical examples are the sidewall surface roughness of a polysilicon etching line and the average grain size for a polysilicon deposition film. These properties are important since they affect directly the quality and performance of the integrated circuit (IC) devices being built. Traditionally, however, they are treated informally and subjectively as tacit knowledge in the processing arena.

A systematic approach to modeling, simulating and controlling such qualitative properties is presented in this work. This approach is based on the statistical analysis of categorical data, and on fuzzy logic theory. The results show significant promise for incorporating qualitative and quantitative features of a process in a computer-integrated manufacturing environment.

Our approach is based on the viewpoint that qualitative process variables are categorical data and can thus be better understood with the help of logistic regression analysis. This analysis reveals important relationships between the input process settings and the qualitative process output responses, in a way that is similar to linear regression analysis for conventional numerical variables. Similarly, categorical process variables can be used for process control, which is driven by a probabilistic model of the categorical variables.

The modeling framework proposed in this work is further built on a self-learning inference system based on fuzzy logic theory. According to this methodology, both the input and output variables (numerical or categorical) are "fuzzified" into linguistic variables, which take several discrete values associated with membership functions. The linguistic values of the input and output variables are mapped through a fuzzy inference process which is implemented as a series of fuzzy set operations. The resulting linguistic values obtained from such inference operations can be "defuzzified" into numerical or categorical values.

Such a fuzzy inference system can be designed and created by training data obtained either from human expert knowledge, or automatically extracted from hard data from statistically designed experiments. After the establishment of the initial inference system, the membership functions can be tuned adaptively to accommodate process changes.

In applying such a qualitative modeling framework, engineers can extract, store, update, and transfer their qualitative understanding and experience about a process along with the quantitative knowledge. Hence, in addition to the conventional statistical, empirical or physical modeling of the processing technology, the qualitative inference systems proposed in this work are also useful in formal process modeling, simulation, and control.

Table of Contents

List of Figures

List of Tables

Acknowledgments

I would like to take this opportunity to express my gratitude to my research advisor, Professor Costas J. Spanos, for his recommendation to choose fuzzy logic and statistical applications on semiconductor manufacturing as my Ph.D. project, and for his continuous inspiration and attentive advice to my study and research work during the past four years. My thanks also go to Dr. K.K. Lin at Intel Corp., for his advice when I started this project, and to Dr. S.W. Butler at Texas Instruments Inc., for her help in developing the statistical analysis for the plasma etching, surface imaging and dry develop processes.

I am grateful to Professor Latfi A. Zadeh for his encouragement and support of my Ph.D. project, and for his being my Thesis Committee member and chairing my Qualifying Exam Committee. I also appreciate his introducing me to participate in the fuzzy logic application project at Hewlett-Packard Laboratory in 1992. I benefited from discussions with Dr. Hideyuki Takagi, Dr. Li-Xin Wang, Dr. Michael Lee and others who belong to the "Fuzzy Logic research group" led by Prof. Zadeh. I would also like to thank other members of my Ph.D. Thesis Committee and Qualifying Committee members: Professor Philip Stark in the Statistics Department and Professor Ping Ko in EECS Department. In particular, Professor Stark's careful review and helpful comments made the statistical aspects of this thesis more rigorous.

The success of my Ph.D. program was also dependent on the cooperation, discussion and friendship with my fellow colleagues in the Berkeley Computer Aided Manufacturing Group (BCAM): Soheila Bana, Bart Bombay, Eric Boskin, Roawen Chen, Sean Cunningham, Zeina Daoud, Haifang Guo, Mark Hatzilambrou, Mehdi Hosseini, Herb Huang,

Sovarong Leang, Sherry Lee, Zhi-Ming Ling, Hao-Cheng Liu, Tom Luan, Shang-Yi Ma, Gary May, Tony Miranda, David Mudie, Fariborz Nadi, David Rodriguez, John Thomson, Pamela Tsai, Eddie Wen, Crid Yu. I specially thank Sherry Lee for her generous help in the last stage of my thesis writing.

I thank Professor Kenneth K. Mei for his advice, support, and understanding of my study and research work in my first year at Berkeley. I enjoyed working with my colleagues in the Electromagnetism group that year: Hua-cheng Chang, Steven Cole, Jia-yuan Fang, Kevin Heppel, Julie Kenrow, Guo-Chun Liang, Yao-Wu Liu, Yong Liu, Rafael Pous and Mark Prouty.

Finally I would like to mention the invaluable friendships I was fortunate enough to enjoy during my years at Berkeley with: Prof.&Mrs. Z. Bai, L. Buckman, Dr. N. Chen, Prof. R. Dong, D. Fraser, Dr.&Mrs. F. Gao, Dr.&Mrs. L. He, Dr. J. Huang, Mr.&Mrs. H. Jones, S.K. Tsui, J. Ouyang, Dr. L. Qin, Dr. S.X. Shen, C. Tung, V.W. Wakil, B. Wu, S.Z. Wu, Dr.&Mrs. Y. Wu, Dr. P.H. Xiao, P. Xu, Dr. J. Yee, K. Yeung, H. Yu, Dr.&Mrs. Y. Yu, Dr.&Mrs. K. Yuan, Dr. W. Yun, Dr.&Mrs. S. Zhou and many other friends of mine.

Last but certainly not the least, I am in debt to my family in China, for their constant support and understanding of my pursuing a great education in U.S. For their love and devotion, I dedicate this dissertation to my parents and sister.

This work was supported by grants from the National Science Foundation (MIP-9014940), Semiconductor Research Corporation (94-YP-700), the California MICRO Program, Texas Instruments Inc., and National Semiconductor Corp.

Chapter 1 Introduction

1.1 Motivation

Process modeling is one of the essential tasks in the semiconductor manufacturing industry today. Extensive research and development has been carried out during the last few years, both in industry and academia [1.1][1.2][1.3]. While some of the modeling approaches target specific processes and equipment, such as Low Pressure Chemical Vapor Deposition (LPCVD) [1.4] or Plasma Etching [1.5], other methods seek to create general modeling frameworks [1.6]. Most of these techniques are focusing on numerical or quantitative process response variables.

In reality, however, many qualitative process observations are being informally collected and used along with quantitative measurements during the development and production runs of IC processes. These observations are mostly used to convey a visual impression: printed patterns are "fuzzy" or "sharp," surfaces are "smooth" or "rough," layers adhere "well" or "badly" to the substrate. To date, these types of observations have been collected side-by-side with quantitative observations, such as line-widths, layer thicknesses, threshold voltages, average film grain size, etc. Even though formal statistical techniques have been used extensively in dealing with quantitative observations, most existing equipment models fail to describe these important qualitative aspects of the process [1.7]. Thus, qualitative observations have remained subjective, and their use has been opportunistic, informal and inefficient. This is especially unfortunate, since qualitative observations capture important information about the process. Further, the collection of

qualitative information usually involves expensive analytical techniques and consumes a great amount of resources [1.8].

The objective of this work is to propose a framework for modeling such qualitative process characteristics by developing an intelligent computer aided manufacturing system that can capture qualitative as well as quantitative aspects of a manufacturing process. Within this system models are developed that describe the qualitative relationships between the equipment settings and the process responses. Statistical analysis methods for categorical data [1.9] and fuzzy logic-based inference theory [1.10] form the mathematical basis for our modeling system, as they have been shown to be effective media for capturing qualitative information.

1.2 Overview

In this dissertation, we introduce a systematic methodology for analyzing qualitative observations in semiconductor manufacturing. Chapter 2 introduces the basics of experimental design [1.11] and categorical data analysis based on logistic regression [1.9]. This regression method will be applied to an actual plasma etching process developed at Texas Instruments, Inc. [1.8]. Chapter 3 proposes a framework of statistical process control (SPC) schemes to monitor and control qualitative process variables based on the logistic regression models and other relevant categorical data analysis tools. This approach is a generalization of the conventional SPC schemes used for numerical process variables.

Chapter 4 presents a brief introduction to fuzzy logic theory, followed by a self-learning inference system based on that theory. A low pressure chemical vapor deposition

(LPCVD) process is used as a test vehicle for such an inference modeling system. And in Chapter 5, a combination of categorical analysis and fuzzy inference is employed to create a framework for an intelligent self-learning modeling system. This method automatically designs an initial fuzzy inference system through statistical categorical analysis of a designed experiment. This initial system can be tuned and adapted to accommodate newly acquired experimental data. This way the system can learn from and adapt to process changes.

Chapter 6 describes the various software programs created during to course of this project. Based on the object oriented programming paradigm, all the programs are written in C, C++ and the X-window tool-kit under the UNIX environment. Chapter 6 also includes a simple introduction to the Berkeley Computer Aided Manufacturing (BCAM) software system and its relationship with this qualitative framework. Additional detail information about these software packages are included in the Appendix. Finally in Chapter 7, potential applications of this work are proposed.

The flow-chart of this dissertation is shown in Figure 1.1, where the interdependence among different chapters is depicted.

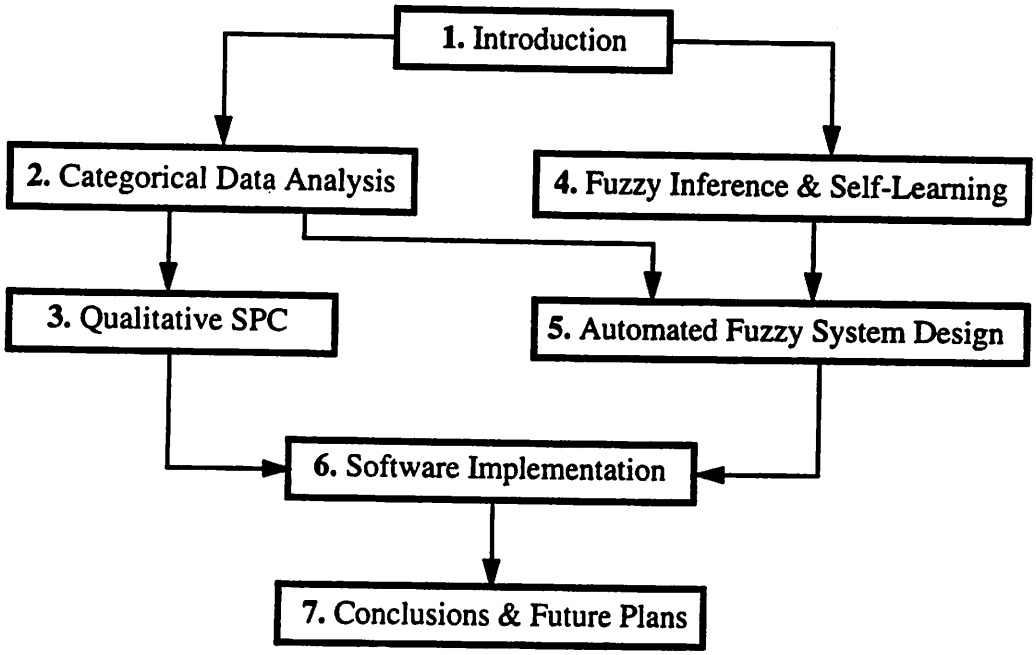


Figure 1.1 Flowchart for This Dissertation

1.3 References

[1.1] P.K. Chatterjee and P.K. Mozumder, "Special Issue on Microelectronics Manufacturing Science and Technology," *IEEE Transactions on Semiconductor manufacturing*, Vol. 7, No. 2, p. 106, May 1994.

[1.2] J. McGehee, J. Hebley, and J. Mahaffey, "The MMST Computer-Integrated Manufacturing System Framework," *IEEE Transactions on Semiconductor manufacturing*, Vol. 7, No. 2, pp. 106-116, May 1994.

[1.3] D.S. Boning and P.K. Mozumder, "DOE/Opt: A System for Design of Experiments, Recipe Surface Modeling and Optimization Using Process and

Device Simulation," *IEEE Transactions on Semiconductor manufacturing*, Vol. 7, No. 2, pp. 233-244, May 1994.

[1.4] E. Sachs, G.H. Prueger, and R. Guerrieri, "An Equipment Model for Polysilicon LPCVD," *IEEE Transactions on Semiconductor manufacturing*, Vol. 5, No. 1, pp. 3-13, February 1992.

[1.5] G.S. May, J. Huang, and C.J. Spanos, "Statistical Experimental Design in Plasma Etch Modeling," *IEEE Transactions on Semiconductor manufacturing*, Vol. 4, No. 2, pp. 83-98, May 1991.

[1.6] D.S. Boning, M.B. McIlrath, P. Penfield, Jr., and E.M. Sachs, "A General Semiconductor Process Modeling Framework," *IEEE Transactions on Semiconductor manufacturing*, Vol. 5, No. 4, pp. 266-280, November 1992.

[1.7] K.-K. Lin and C.J. Spanos, "Statistical Equipment Modeling for VLSI Manufacturing: An Application to LPCVD Reactors," *IEEE Transactions on Semiconductor Manufacturing*, Vol.3, No.4, pp.216-229, November 1990.

[1.8] S.W. Butler, Texas Instruments, Dallas, Texas, *private communication* (summer and fall of 1992)

[1.9] D. W. Hosmer and L. Lemeshow, *Applied Logistic Regression*, New York: John Wiley, 1989.

[1.10] H.-J. Zimmermann, *Fuzzy Set Theory - and its Applications, Kluwer Academic Publishers* (1991).

[1.11] G.E.P. Box, W.G. Hunter and J.S. Hunter, *Statistics for Experimenters - An Introduction to Design, Data Analysis, and Model Building*, John Wiley & Sons, 1978.

Chapter 2 Categorical Data Analysis

According to established statistical nomenclature, a variable can be *interval* or *categorical.* An interval variable is one that can take continuous numerical values. A categorical variable, on the other hand, takes only discrete values. If the values can be ordered (such as the linguistic values "smooth," "rough," "very rough"), then the variable is an *ordinal* categorical variable. If no ordering is possible (such as the linguistic values "green," "blue," "red"), the variable is a *nominal* categorical variable [2.1][2.2].

Categorical variables are encountered frequently in semiconductor manufacturing processes, but are often treated informally. For example, in plasma etching, the roughness of the etched polysilicon side-wall surface is an important indicator of the quality of an integrated circuit (IC) product. This quality depends on various process input settings that comprise the process recipe.

Statistical analysis of categorical variables is used extensively in biological and social sciences, but has not been applied to semiconductor manufacturing. For example, there might be a relationship between a person's age (an interval input variable) and the *likelihood* to develop heart diseases, an ordinal categorical variable that may take discrete values such as *not likely*, *quite likely*, or *very likely.* While the most common modeling technique for an interval variable is linear regression analysis, a popular modeling method for categorical data is logistic regression [2.1].

2.1 Background Knowledge — A Plasma Etching Process

A semiconductor plasma dry develop (through etching) process developed at Texas Instruments, Inc. (TI) [2.3] is used as the test vehicle for developing the qualitative modeling methodology in this work. The objective of the etching process is to create polysilicon lines matching a target pattern as close as possible. The process settings are the temperature, etch time, power, pressure and oxygen flow rate inside the reactor. Experimental data were obtained from a factorial design [2.4] based on these inputs. Many process output variables are important in evaluating such a process. These output variables include not only quantitative variables such as *average line width* of a printed pattern and the *uniformity* of the line width across the wafer, but also qualitative variables such as the sidewall *roughness*, and the presence of indentations in the photoresist line profile, called *mouse bites* (Figure 2.1).

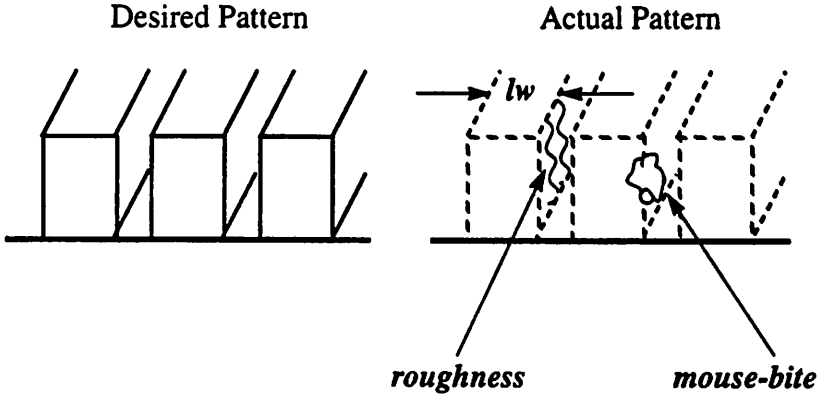


Figure 2.1 A Plasma Etching Process

These ordinal categorical output variables take only a few discrete values, usually labeled as "very rough," "rough," "smooth," "very smooth," *etc.* These values are assigned by trained human operators who use established references and inspect the etched wafer through a scanning electron microscope (SEM).

2.2 Statistical Regression Analysis

In this section we will discuss both linear and logistic regression methods, which can be used to analyze continuous and categorical output variables, respectively.

2.2.1 Linear regression modeling

In a single input, single output process, when both the input variable x and the output variable y of a process are continuous interval variables, a linear regression model can be applied, if we assume there is a linear relationship between x and y. This linear model is expressed as $y_n = \alpha+\beta x_n+\varepsilon_n$, where n is the nth observation for the variables x and y, and ε is the error. Statistically, all the experimental runs are assumed to follow the above linear model, subject to identically, independently and normally distributed errors, ε_n. If a and b are the estimates of the model parameters α and β, respectively, then the estimated linear model can be expressed as [2.4],

$$y_n = a + bx_n + e_n = \hat{y}_n + e_n, \qquad n = 1, 2, ..., N \tag{2-1}$$

where the subscript n stands for the nth experimental run, y_n is the measured experimental value, $\hat{y}_n = a + bx_n$ is the respective predicted value, and e_n is the *residual*. The parameters a and b are typically estimated through the method of least squares, *i.e.* they are chosen to minimize the sum of the squares of the residual e_n. Eq. (2-1) is called *regression equation* and the output response variable y is sometimes called the *dependent variable*, while x is called the *independent variable* or the *regressor*. When more than one input is considered, the term bx_n in Eq. (2-1) should be replaced by a vector product $\boldsymbol{b}^{\mathrm{T}}\boldsymbol{x}_n$, where $\boldsymbol{x}$ represents the input vector in a multi-dimensional space, and $\boldsymbol{b}$ is the corresponding parameter vector. In this case, the first equation in Eq. (2-1) represents a surface in a multi-

dimensional space, so sometimes this model is also referred as *response surface model.* An analysis of variance table (ANOVA) can be used to test the significance of each contributing factor, as well as the overall significance of the model. Lack-of-fit tests can be used to determine if additional terms or transformations are needed [2.4][2.5].

2.2.2 Logistic regression modeling

The output variables of a process are often categorical data such as in the case of mouse-bites and surface roughness mentioned above. Categorical variables take values from an ordered set such as {"excellent," "good," "fair," "poor," "worst"}. In this case, the conventional linear regression model would not be very useful. On the other hand, when a semiconductor manufacturing process is under statistical control, we can assume that these categorical variables take their discrete values with fixed probabilities, and that they can be observed through a systematic and consistent measurement procedure. We also assume that when the process is in control, the category in which a given output "lands" is independent of prior observations. This amounts to assuming that in n experimental trials with the same process input parameters, the number of outputs in each category will be jointly multinomially distributed. Further, the probabilities of the different categories will be functions of the input parameters; in particular, we assume that the odds in favor of category "i" will have the form indicated in Eq. (2-3), shown on the following page. Models of this form can be fitted with logistic regression [2.1][2.2]. As will be seen later, that functional form and its sigmoidal shape can easily be translated to fuzzy membership functions. Further, these logistic models have a statistically adequate fit to the data.

Such *logistic regression* or *logit* model, creates a mapping of the continuous input vari-

ables to the *probabilities p* that the output variable fall in each category, *i.e.*, the logistic method estimates the probability p_i of having results in category i, i=1,...,m. In order to describe this model, we will first define the *cumulative probabilities* π_i for a certain value[1] of x as follows:

$$\pi_i(x) = \sum_{j=1}^{i} p_j(x) \tag{2-2}$$

The procedure of applying this method is summarized by the formula that defines the logistic fitting curves:

$$\pi_i(x) = \frac{\exp(a_i + b \cdot x)}{1 + \exp(a_i + b \cdot x)} \quad i = 1, 2, \ldots, m-1 \qquad \text{where } a_{i-1} < a_i \tag{2-3}$$

where π_i are the cumulative probability functions. From this, the actual probabilities can be estimated as:

$$p_i(x) = \begin{cases} \pi_1(x), & i = 1, \\ \pi_i(x) - \pi_{i-1}(x), & i = 2, \ldots, m-1, \\ 1 - \pi_{m-1}(x), & i=m \end{cases} \tag{2-4}$$

Similarly to the linear regression case, a and b are the estimated parameters, and x is the input variable. Since the output y is a categorical variable, $p_i(x)$ gives the probability

1. "Cumulative probability" in this work is the probability which accumulates over the categories of the categorical variable x.

that y falls into *category* i (i=1, 2, ..., m). As in the linear regression case, various tests can be applied to the estimated parameters and the significance level of the departure of the corresponding input variable from zero can be evaluated.

Figure 2.2 shows an example of what the logistic fitting functions and the probability functions might look like, when plotted against one of the continuous explanatory variables. As we see in Eq. (2-3), the cumulative probability curves are distinguished from each other through the different constant parameters a_i. In reality, when two categories are not significantly different we can combine them to be the same category without degrading the fit.

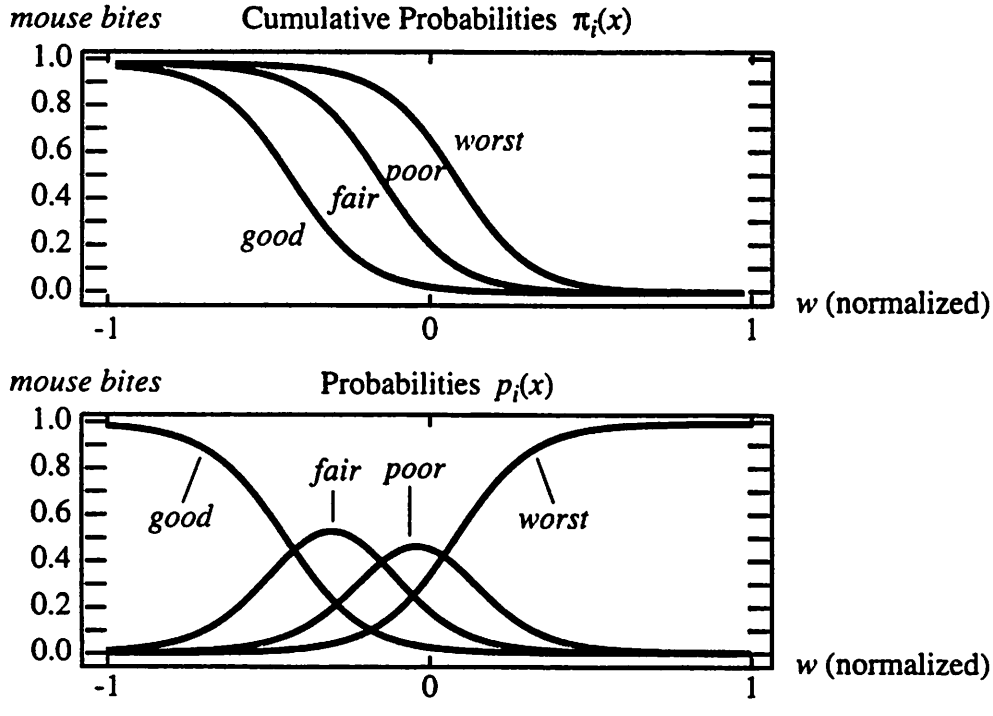


Figure 2.2 Logistic Regression Results — Mouse Bites Probabilities vs. Power w

2.3 Design of Experiment

In order to understand the relationship between certain output response variables and the input process settings, a set of experiments can be run to obtain the data needed. The most efficient way to carry out such an experiment is to set the input settings in some particular arrangement so that the effects of these input variables can be determined through a minimal number of experiments. The systematic approach to conducting such an investigation effectively is called *statistical experimental design* [2.4].

One of these approaches is the factorial design, where the experimental settings are chosen to be at the corners of a multi-dimensional *cube* in the input space. Figure 2.3(b) depicts a 2-dimensional example. Comparing the more intuitive alternative way of changing one input parameter at a time, the benefit of factorial design is seen clearly. If the four points in (a) are chosen, the effect for x_1 and x_2 would be estimated as $y_d - y_a$ and $y_b - y_c$, respectively. However, if the four points in (b) are chosen, these two effects will be $\frac{(y_3 + y_4)}{2} - \frac{(y_1 + y_2)}{2}$ and $\frac{(y_2 + y_4)}{2} - \frac{(y_1 + y_3)}{2}$, respectively. It can be shown [2.4] that option (b) gives better precision with the same number of experiment runs. In addition, option (b) allows us to estimate the interaction $\frac{(y_1 + y_4)}{2} - \frac{(y_2 + y_3)}{2}$ of the two inputs.

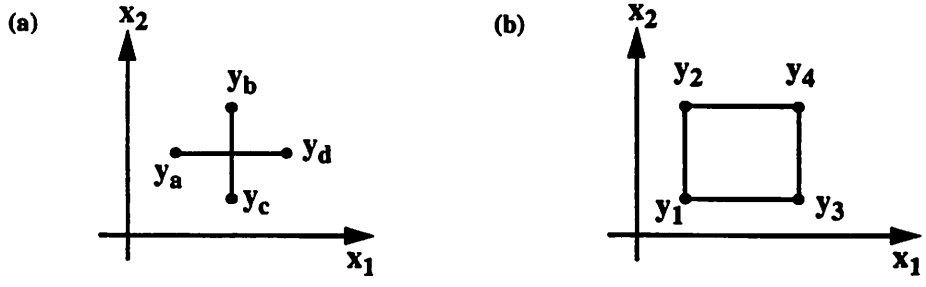


Figure 2.3 Factorial Experimental Design in a Two-Dimensional Space

There are many variations of factorial designs, and many other approaches to experimental design besides factorial design. The experimental data used in this work are based on the plasma etching process mentioned before, and these experiments were carried out at TI using a 3-level 3-factor factorial design (see Section 2.4).

2.4 Analyzing the Qualitative Experimental Data — Mouse Bites

Table 2.1 shows the experimental data collected from a plasma etching process at Texas Instruments, Inc. (To protect proprietary information, only the normalized values {-1, 0, 1} of the input parameters are shown). This is a full 3-level factorial experiment with 6 replicates at the center.

Table 2.1 A Surface Imaging, Dry Develop Process (*Courtesy*: Texas Instruments)

No.	Power (w)	Pressure (p)	Oxygen Flow (O_2)	*roughness* {smooth, fair, rough, roughest}	*mouse bites* {good, fair, poor, worst}
1	1	-1	1	roughest	poor
2	-1	0	-1	fair	fair
3	0	0	0	fair	poor
...	...	...	...	...	...
29	-1	-1	0	smooth	good
30	-1	1	-1	fair	fair
31	0	-1	-1	rough	worst
32	0	0	0	rough	fair

The model for the mouse bites used four levels based upon the logistic statistics. It is used in this section to illustrate how to carry out logistic regression to analyze categorical process variables such as surface *roughness* and *mouse bites*. The commercial software used is called JMP [2.6]. JMP is an interactive subset of SAS and runs on Macintosh computers.

Table 2.2 to Table 2.5 contain the results of the logistic analysis to assess the logit model for mouse bites versus all three input factors (w, p, and O_2). The guiding principle in logistic regression is the same as in linear regression, namely, to compare the observed values of the response variable to the model predicted values *with* and *without* the input variable in question. In the linear regression model, this evaluation can be realized simply through the sum of squares of the residuals with and without considering the input variables. In the logistic regression model, however, the corresponding estimator is the so called Log Likelihood function, which is a measure of the difference between the observed values and the model prediction values, with or without considering input variables. The comparison to assess a logit model is carried out though the following formula:

$$G = -2\ln\left[\frac{\text{(likelihood without the variable)}}{\text{(likelihood with the variable)}}\right] = -2\ln\left[\frac{\prod_{k=1}^{M}\left(\frac{n_k}{n}\right)}{\prod_{k=1}^{M}\left(\prod_{i=1}^{n}\pi_k(i)^{y_k(i)}\right)}\right] \tag{2-5}$$

where $\pi_k(i)$ is the cumulative probability for category k for the i^{th} sample, n_k is the number of samples falling into category k, and $y_k(i)=1$ when the i^{th} sample falls into category k and $y_k(i)=0$ otherwise. Also in Eq. (2-5), $n = \sum_k n_k$, $n_k = \sum_i y_k(i)$ and $k=1,2,...,M$, where M is the total number of categories. Such a likelihood ratio G obeys asymptotically the Chi Square (χ^2) distribution with M-1 degrees of freedom [2.1]. From the result showing in Table 2.2, we see that the χ^2 value for the model is 40.879 and the probability that this or a larger χ^2 occurs by mere chance is very close to zero, indicating the strong significance of

the model.

Table 2.2 Model Fit for Mouse Bites (Including w, p, and O_2)

Statistic	w/o variables	w/ variables	Chi-Square	Probability
-2 Log Likelihood	83.075	42.196	40.879	< 0.0001

Table 2.2 gives a detailed list of the estimated parameters for the logit model. As we see from the table, not every input factor is significant. For instance, the χ^2 value for O_2 flow rate is only 2.4 and the corresponding significance level is more than 12%. Accordingly, we can eliminate the insignificant input factor O_2 from our mouse bites logit model and re-compute the logit model analyses.

Table 2.3 Logit Coefficients and Effect Test (w, p, and O_2)

Coefficients	Value	Std. Error	Chi-Square	Probability
Intercept 1 (α_1)	-4.6645	1.3184	12.52	0.0004
Intercept 2 (α_2)	-1.7653	0.7084	6.21	0.0127
Intercept 3 (α_3)	0.6485	0.5902	1.21	0.2719
Power w (β_1)	-5.2884	1.3620	15.08	0.0001
Pressure p (β_2)	-1.3700	0.6680	4.21	0.0403
Oxygen Flow Rate O_2 (β_3)	0.9962	0.6425	2.40	0.1210

The results of the logistic modeling excluding O_2 are shown in Table 2.4 and Table 2.5. From comparing Table 2.2 with Table 2.4, and Table 2.3 with Table 2.5, we see the simpler model without O_2 is just as good as the previous model with all three inputs.

Table 2.4 Model Fit for Mouse Bites (Including only w and p)

Statistic	w/o variables	w/ variables	Chi-Square	Probability
-2 Log Likelihood	83.075	44.910	38.165	< 0.0001

By convention, a variable is considered significant if its significance probability is less than 5% [2.4][2.7]. We see from Table 2.3 or Table 2.5 that both inputs w and p are significant according to our model, however the power w is a much more dominant factor, as its χ^2 value is much larger than that of the pressure p.

Table 2.5 Logit Coefficients and Effect Test of the Reduced Model (w and p)

Coefficients	Value	Std. Error	Chi-Square	Probability
Intercept 1 (α_1)	-4.3228	1.2022	12.93	0.0003
Intercept 2 (α_2)	-1.7561	0.6895	6.49	0.0109
Intercept 3 (α_3)	0.5915	0.5689	1.08	0.2985
Power w (β_1)	-4.833	1.2325	15.38	0.0001
Pressure p (β_2)	-1.2598	0.6318	3.98	0.0461

Figure 2.4 plots the different models for mouse bites. The probability curves agree well with results from the above tables. There is almost no visible difference between the full model (a) and the reduced model (b) since input O_2 is insignificant. There are small visible differences between (b) and a model (c) based solely on power w, since p is significant but not as much as the main factor w. For comparison, a model (d) based solely on pressure p is also shown.

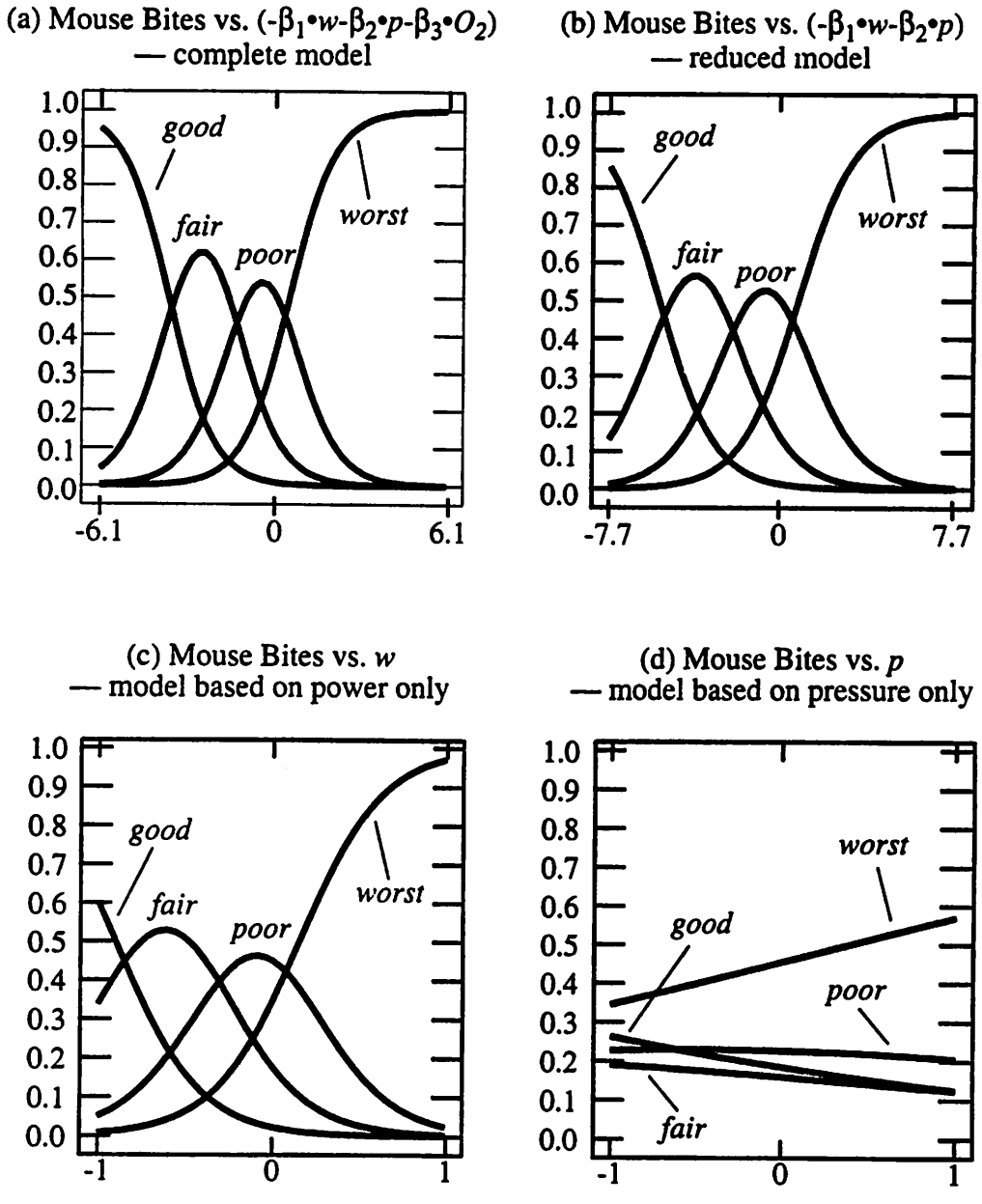


Figure 2.4 Probability Plots for Mouse Bites — Four Possible Models

This procedure of screening the insignificant input factors is important in reducing the complexity of a modeling system. In reality, a process may have a large number of input factors, but it is too costly and (as we have seen in the above example) unnecessary to include all inputs in modeling a process response.

2.5 Summary

Many qualitative characteristics in a semiconductor manufacturing process can be treated as statistical categorical data. Similar to linear regression for analyzing conventional numerical data, a logistic regression model can be applied to these qualitative process variables collected along with the quantitative data from designed experiments. This regression modeling technique can reveal relationships between these qualitative process outputs with input process settings such as power, pressure and Oxygen flow rate in a typical plasma etching process. This analysis can also be used to screen out insignificant input factors in modeling the response of a process.

2.6 References

[2.1] D.W. Hosmer and S. Lemeshow, *Applied logistic regression*, John-Wiley & Sons, 1989.

[2.2] A. Agresti, *Analysis of Ordinal Categorical Data,* New York: John Wiley, 1984.

[2.3] S.W. Butler, Texas Instruments, Dallas, Texas, *private communication*, summer and fall of 1992.

[2.4] G.E.P. Box, W.G. Hunter and J.S. Hunter, *Statistics for Experimenters - An Introduction to Design, Data Analysis, and Model Building*, John Wiley & Sons, 1978.

[2.5] R.F. Gunst and R.L. Mason, *Regression analysis and its application*, M. Dekker, New York, 1980.

[2.6] JMP Manual, SAS Institute Inc., SAS Campus Drive, Cary, NC 27513.

[2.7] R.V.Hogg and E.A.Tanis, *Probability and Statistical Inference*, Macmillan Publishing Company, New York, 1988.

Chapter 3 Statistical Process Control Based On Categorical Data

In Chapter 2, logistic regression models are created for categorical process outputs. Once these models have been shown to represent the process accurately, they can be used for quality control during production. These "model-based" Statistical Process Control (SPC) schemes allow the controllable variables of a process to change during production. The model is then used to account for these changes during SPC. This chapter proposes several different SPC schemes for categorical variables.

As defined, the logit model predicts the logs of cumulative probabilities of several output categories as a linear function of the controllable process inputs [3.1]. For SPC, however, the actual probabilities of each category can be assessed. These probabilities can be calculated by taking consecutive differences between the cumulative probabilities, as shown in Eq. (2-4). There is usually a significant prediction error in these probabilities, since the model is based on limited sampling of noisy experimental data. This is in contrast to standard SPC practice, where the classical $\pm 3\sigma$ control limits are considered "exact." The probabilities of each category cannot be considered exact because in a typical categorical modeling situation the sample size is limited and the confidence interval of the predicted probability values is wide. The error in probability prediction can be estimated with the assumption that the estimated model parameters have a multivariate normal distribution.

3.1 Short-Term SPC Using Categorical Models

The objective of short-term SPC is to identify abrupt (*i.e.* short term) process malfunctions during production. Since the logistic models are made to predict probabilities, the corresponding SPC control procedure is based on the predicted probabilities of each outcome category of a process. For a certain category, the prediction error is usually substantial and is calculated first.

The goal is to find the 67% (i.e. $\pm\sigma$) confidence intervals of the predicted probabilities [3.2]. The objective is to start with the multivariate normal distribution of the intercepts and the coefficients, and to transform this distribution as an expression of the linear combinations of the three input variables: power w, pressure p, and Oxygen flow rate O_2. Mathematically, this transformation is equivalent to a linear change of coordinates in a multi-dimensional space. Once such a transformation is accomplished, the linear combinations of the three input variables can be set to their correlated $\pm\sigma$ extremes in order to calculate the extreme predicted probabilities for each category.

Consider, for example, the probabilities of each category of mouse bites, at the nominal settings of the process (Figure 3.1). The depicted 67% confidence intervals in Figure 3.1 are calculated from the variance-covariance matrix of the estimated model parameters. This is accomplished by transforming the multivariate normal statistics of the model parameters to the multivariate normal statistics of the linear combinations of the process parameters and the intercepts. For SPC purposes, only the category that has very small probability (category 2 in this example) needs to be examined.

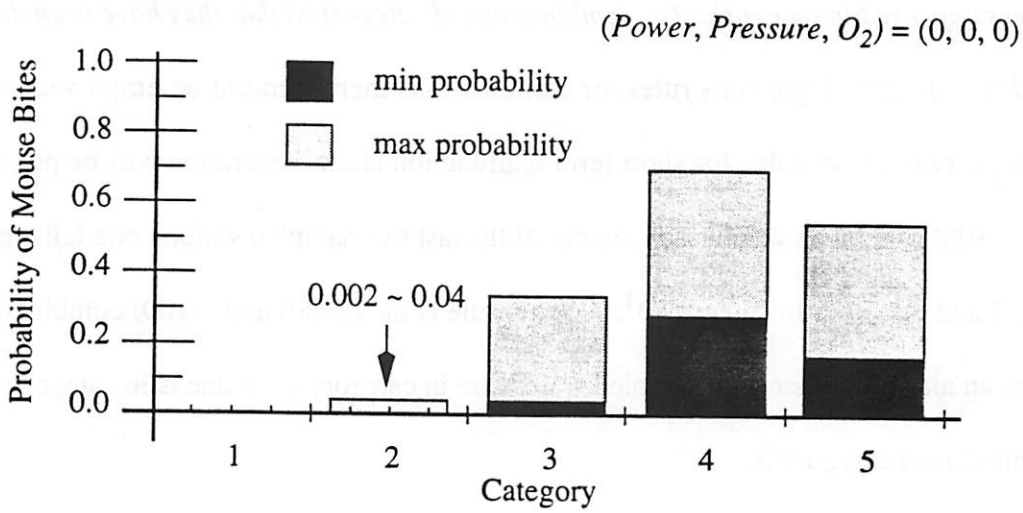


Figure 3.1 Example of Category Histograms with 67% Confidence Intervals.

There are two main objectives in a statistical process control scheme. One is to reduce the probability of "false" alarms. This probability is also called *type I error* rate, and is symbolized by α. The other objective is to reduce the probability of missing the "true" alarms, or the rate of *type II error*, symbolized by β. This objective is equivalent to increasing the probability of generating "true" alarms, or equivalently, increasing the *power of the test*, $1-\beta$.

Another way to explain this concept is to discuss the average run length (ARL) of a control chart. The ARL is defined as the average number of runs before an out-of-control alarm occurs [3.3]. Therefore, if the process is in control, $ARL_0 = 1/\alpha$. Similarly, if the process is out of control, $ARL = 1/(1-\beta)$. Ideally we want a very long ARL_0 and a very short ARL.

In order to evaluate the ARL, one must define the rules for alarm generation. In this

work we will consider *sequential* runs rules, *i.e.* "*issue an alarm if k out of the last n wafers appeared in categories (or combinations of categories) that they have been deemed unlikely.*" Either single runs rules, or combinations thereof might be employed. In this example two sets of rules for short term malfunction alarm generation will be presented. The (1100) rule raises an alarm when out of the last two sampled wafers, one falls in category 2 and the other in category 3[1]. A better rule is the (2000) and (1100) combination: it raises an alarm if either both sampled wafers are in category 2, *or* one is in category 2 and the other is in category 3.

The ARL corresponding to these runs rules are calculated based on the multinomial distribution:

$$P\{x = (x_1, x_2, ..., x_k)\} = k! \prod_{i=1}^{k} \left(\frac{\pi_i^{x_i}}{x_i!} \right) \tag{3-1}$$

The resulting ARL curves are plotted in Figure 3.2 against the actual probability of category 2 in a malfunctioning process. By definition, the probability of a false alarm is the inverse of ARL_0, and the maximum false alarm probability within the range of prediction of the model is the inverse of $min(ARL_0)$, which is plotted in Figure 3.2.

1. Category 1 was not included in this analysis since it was not represented in the experimental data.

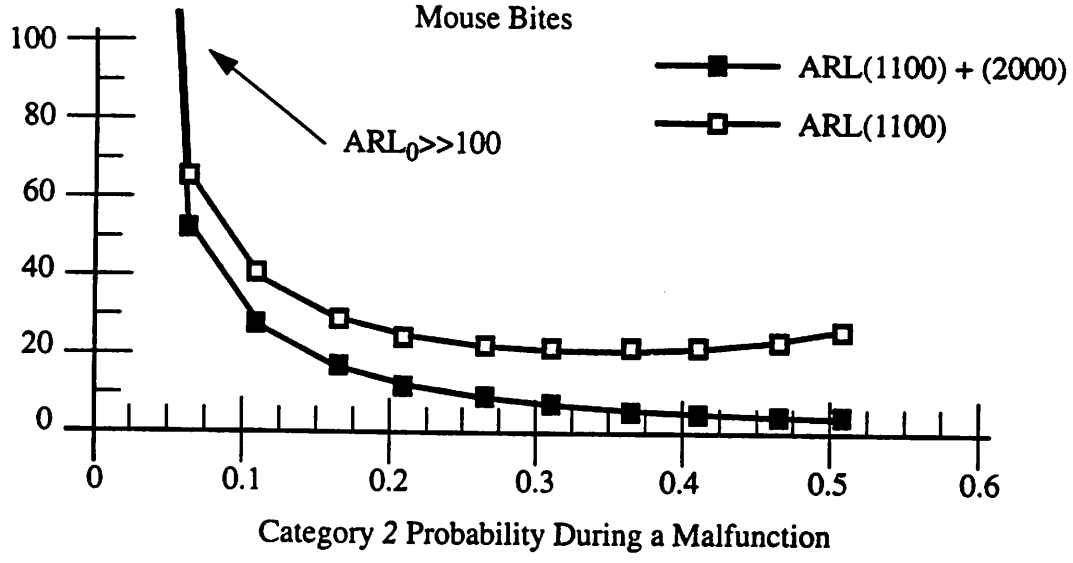


Figure 3.2 ARL for Various Rules as a Function of the Category 2 Probability.

What follows is another example of short-term SPC applied to the process optimized for mousebites. Now category 5 is the least likely category, and it will be used for alarm generation. A single runs rule will be sufficient here: raise an alarm if one of the last two sampled wafers is in category 4 and the other is in category 5. The rule delivers nice ARL characteristics, with relatively low probability of false alarms, and high sensitivity to moderate shifts of the process.

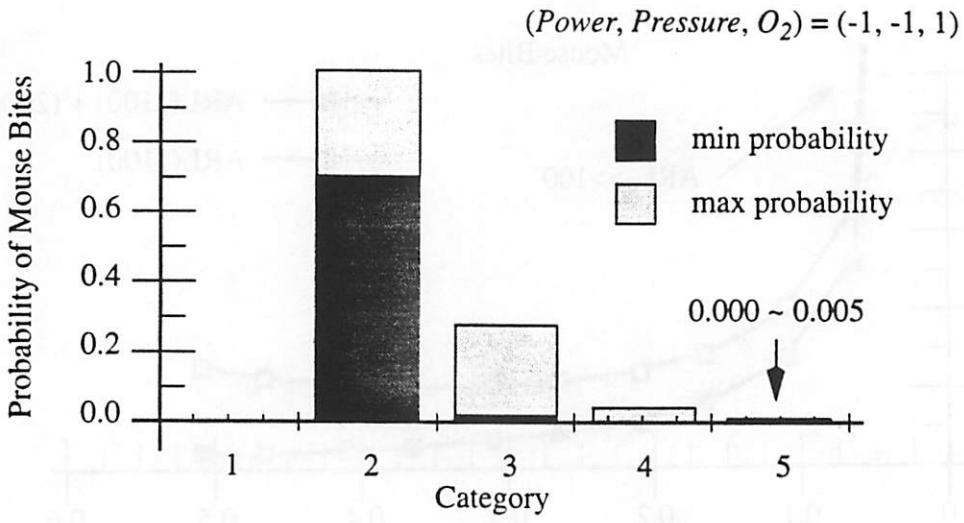


Figure 3.3 Predicted Histogram and Probability Ranges for the Optimized Process

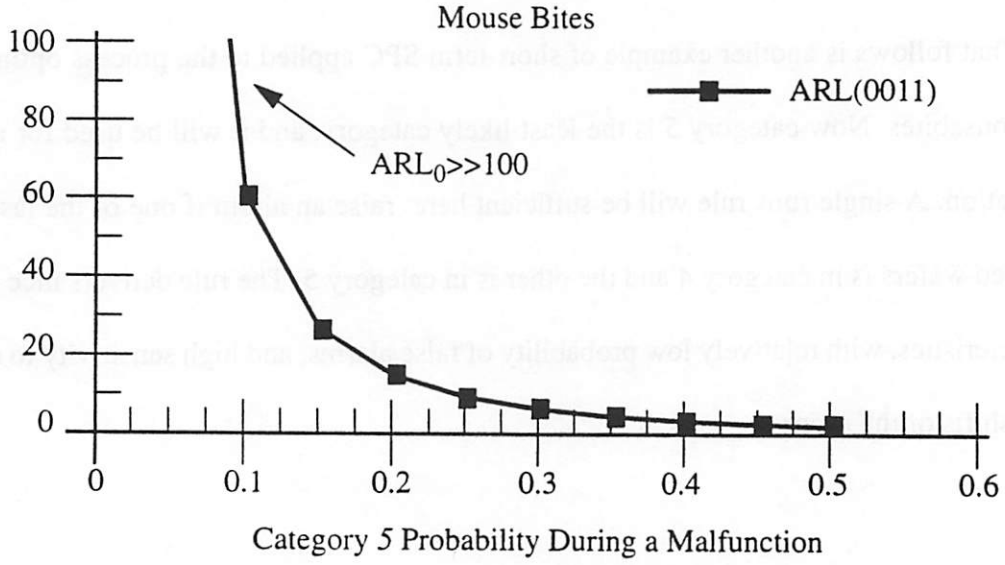


Figure 3.4 Characteristic Function of *0011* Runs Rule Applied to the Optimized Process

In summary, the characteristics of the short-term SPC show that it is possible to generate reliable alarms, as long as we can find either individual or combinations of low-probability categories.

3.2 Long Term SPC

In addition to abrupt malfunctions in the process, there will be slow (i.e. long term) process drifts that might happen due to natural and unavoidable process aging. These might be caused by depletion of chemicals, seasoning of the chamber, scheduled or unscheduled maintenance events, *etc.* A different approach can be adopted to catch such slow process drifts, since, unlike short term alarms, slow drifts maybe be corrected by simple feedback adjustments. The number shown in (3-2) is called *power divergence*, a cumulative figure of merit that describes the match between the process and its model:

$$\sum_{i=1}^{k} \frac{\left(\Pi_i - \hat{\Pi}_i\right)^2}{\hat{\Pi}_i} \tag{3-2}$$

where Π_i is the observed probability and $\hat{\Pi}_i$ is the value predicted by the model for the last k observations. The power divergence observes the Pearson chi-square statistic and is (approximately) distributed according to a known χ^2_{k-1} distribution [3.4]. Therefore an upper control limit can be found for this number for an acceptable false alarm level. If the limit is exceeded, then this long-term SPC model will indicate an alarm. Furthermore, as will be discussed in Section 3.3, this alarm means that the model is no longer consistent with the process, and it has to be re-fitted. A process control action (recipe update) may follow each adaptation of the model.

In this section, two examples of long-term SPC are examined. In the following simulated examples, the Pearson chi-square statistic is used as a function of time, given a simulated drift in the process.This drift of the process is simulated by shifting the *expected*

probability values shown in Eq. (3-2) (See Figure 3.5).

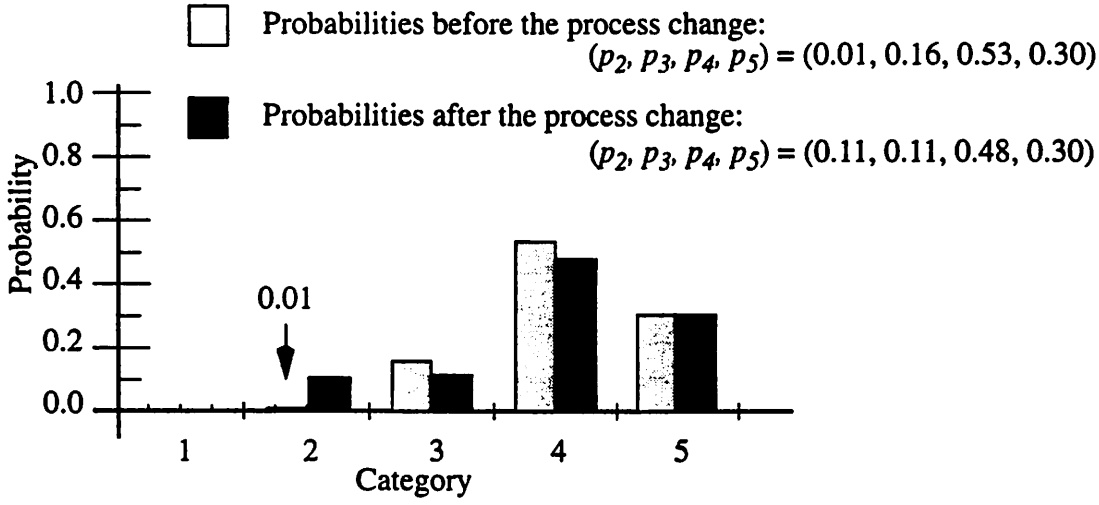


Figure 3.5 Category Probabilities Before and After the Process Change

Such a model-based SPC scheme can be applied even if the controls of the process are allowed to move. In the first example, the process is *locked*, *i.e.* the controls of the process (input vector or recipe setting x) are not changing during the simulated runs. In the second example, however, all the controls of the process are allowed to change randomly within the center 50% of their experimental range in order to simulate a situation where the process is constantly being modified by a run-to-run controller.

The power divergence statistic is calculated over a window of the last 40 runs. The change depicted in Figure 3.5 is introduced at wafer #100. The long-term SPC control charts results for the Power-Divergence statistic, as defined in Eq. (3-2), are shown in Figure 3.6 and Figure 3.7, for the locked and unlocked processes, respectively. The change introduced for an unlocked process (Figure 3.7) is similar to that of the locked process, except there are no constant probability distributions for the four categories.

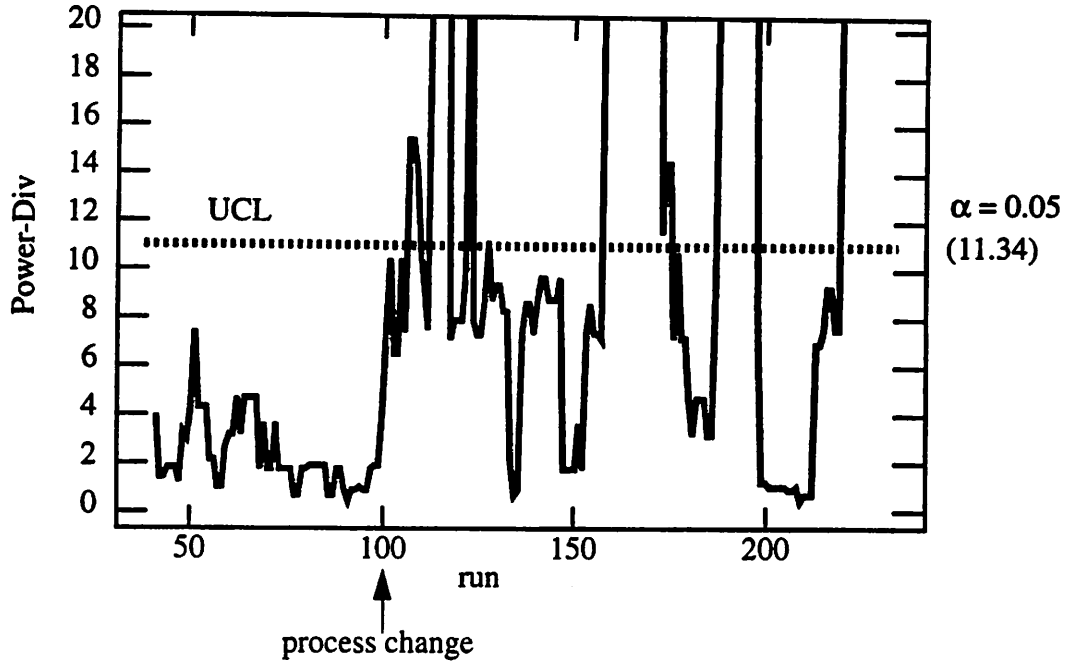


Figure 3.6 A Long-Term SPC Example with Locked Process Settings

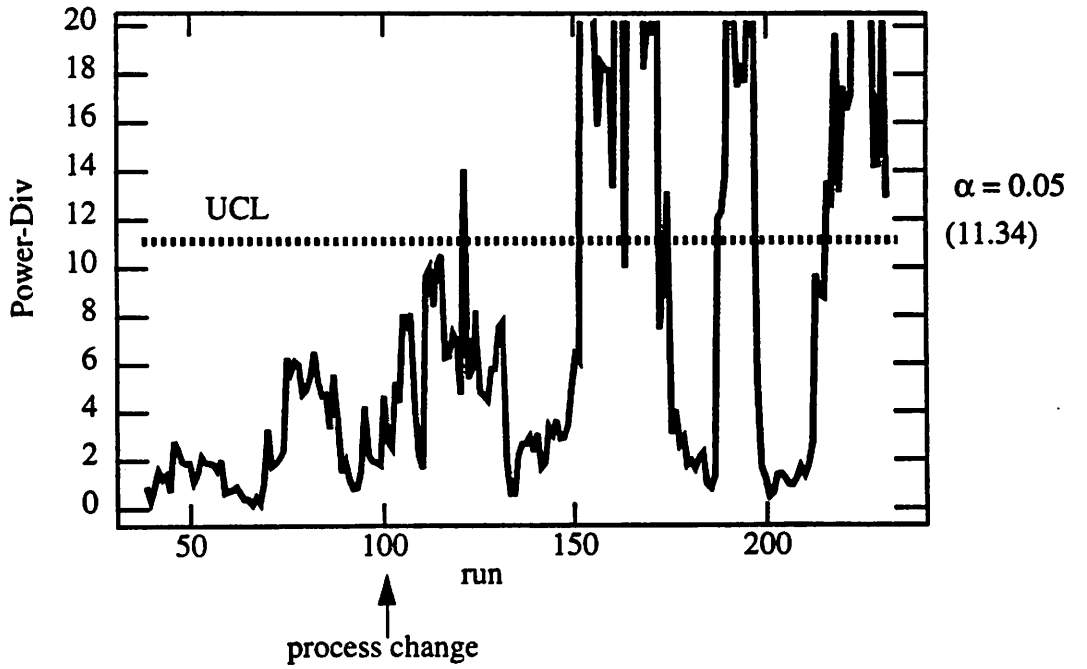


Figure 3.7 A Long-Term SPC Example with Changing Process Settings

3.3 A Model-Update Algorithm for Process Control

Let us assume that a process is in control most of the time. Thanks to the short term SPC scheme we discussed earlier, an abrupt change in the process would cause an out-of-control alarm. For a longer period of time, however, there might be a permanent change or shift in the process settings and the machine conditions. When such a change in the process is detected by the long term SPC scheme, the model must be updated in order to describe the "new" process. This can be accomplished by readjusting some of the intercepts, while using the same linear combination of process parameters. This model update is implemented using a maximum likelihood approach, based on a weighted least squares minimization scheme, which will be illustrated in this section.

3.3.1 Model update approach

Probability prediction models are usually fitted by maximizing the likelihood of the observed data. This maximization is done by choosing the appropriate values of the model used to describe the data. In the case of long term SPC model update for categorical data, each observation contributes some *Log Likelihood*, defined as the weighted sum of all the prediction probabilities over the training set. Therefore, the fitting problem can be recast as a minimization problem:

$$\min\left(-2\mathrm{Log}L\right) \Rightarrow \min\left(-2\sum_{j} w_j \log\left(\hat{p}_j\right)\right) \tag{3-3}$$

where w_j is the weight of the j^{th} observation, and $\hat{P}_j$ is the estimated probability. The model that yields the highest probabilities for the actual observations "fits" the best.

If the observations are collected sequentially in time, then they can be pre-weighted using an exponentially weighted moving average (EWMA) scheme that assigns less importance to older observations. The steepness of the EWMA scheme can be adjusted to match the dynamics of the process. The weighting used in our example is shown in Figure 3.8.

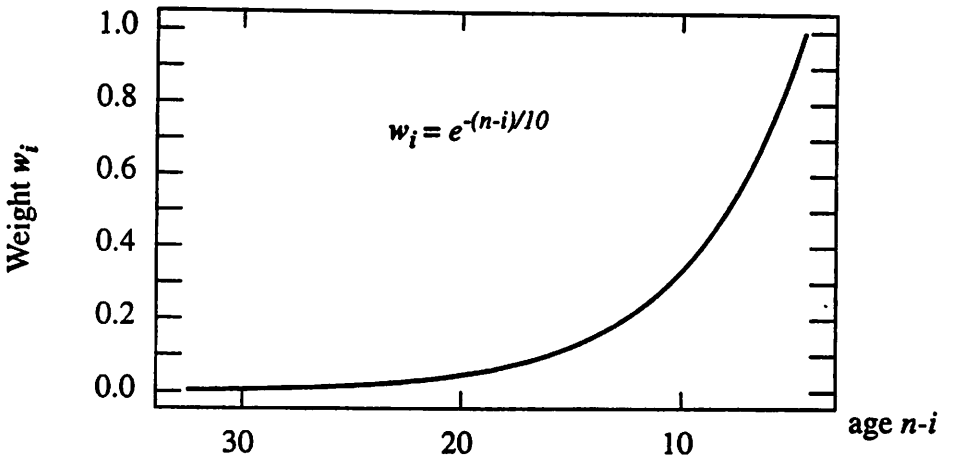


Figure 3.8 EWMA Modeling Update

The general way of fitting a probability model is accomplished by maximizing the *likelihood* of the observation vector. This leads to a numerical minimization problem that can be solved by using the Gauss-Newton iterative method depicted here. The method of "Iteratively Re-weighted Least Squares (IRLS)" [3.5] is employed by solving the following equation for the multinomial variable vector $\underset{\sim}{Z}_j$:

$$\sum_j D_j^T W_j (\underset{\sim}{Z}_j - \underset{\sim}{p}_j) = 0 \qquad (3\text{-}4)$$

where $\underset{\sim}{p}_j$ is the vector of category probabilities computed for the current model parameters. D_j is the matrix of partial derivatives of $\underset{\sim}{p}_j$, and W_j is the weight matrix. The actual solution is found iteratively by the Gauss-Newton method:

$$\hat{\underline{\gamma}}_{m+1} = \hat{\underline{\gamma}}_m + \left(\sum_j \hat{D}_j^T \hat{W}_j \hat{D}_j\right)^{-1} \sum_j \hat{D}_j^T \hat{W}_j (\underline{Z}_j - \hat{\underline{p}}_j) \tag{3-5}$$

where $\hat{\underline{\gamma}}_m$ is the model parameter vector (a1, a2, ..., ak, b) after the mth iteration. $\hat{D}_j$, $\hat{W}_j$ and $\hat{\underline{p}}_j$ are approximations to D_j, W_j and $\underline{p}_j$, evaluated at $\hat{\underline{\gamma}}_m$. The weights are adjusted before each iteration, so that observations that do not fit very well are deemphasized. In general, the weight of an observation is the inverse of its estimated variance at each iteration.

3.3.2 Examples of model adaptation

As in the case of long-term SPC, a process shift will be introduced as depicted in Figure 3.5. Note that for the unlocked process, the probability of each category is no longer constant, but the change still follows the formula in Figure 3.5.

Figure 3.9 shows an example of the locked (fixed input) process where the "routine" process experiences a moderate shift after wafer #100. The change is detected by the long-term SPC algorithm and the update algorithm readjusts the intercepts of the model. The long-term SPC alarm was generated at wafer #130 and the model was updated at wafer 160. Notice that after the model adaptation, the value of the power-divergence statistic drops below the control limit. Eq.(3-6) shows the model change due to the difference of parameter a_1 before and after the model adaptation at wafer #160.

$$\pi_1(x) = \begin{cases} \dfrac{\exp(4.6644506 - \boldsymbol{b} \cdot \boldsymbol{x})}{1 + \exp(4.6644506 - \boldsymbol{b} \cdot \boldsymbol{x})} & \text{wafer \#1 to \#159} \\ \dfrac{\exp(0.730513 - \boldsymbol{b} \cdot \boldsymbol{x})}{1 + \exp(0.730513 - \boldsymbol{b} \cdot \boldsymbol{x})} & \text{after wafer \#160} \end{cases} \tag{3-6}$$

Note that all three intercepts change in order to accommodate the shift in the cumulative distributions (see Table 3.1).

Table 3.1 A Model Adaptation Example

change	before	after
	(p_2, p_3, p_4, p_5)	$(p_2 + .1, p_3 - .05, p_4 - .05, p_5)$
linear term: $b \cdot x$	$-5.288\, w - 1.370\, p + 0.996\, O_2$	
intercept: α_2	-4.6645	-0.7305
intercept: α_3	-1.7653	0.0965
intercept: α_4	0.6485	2.1631

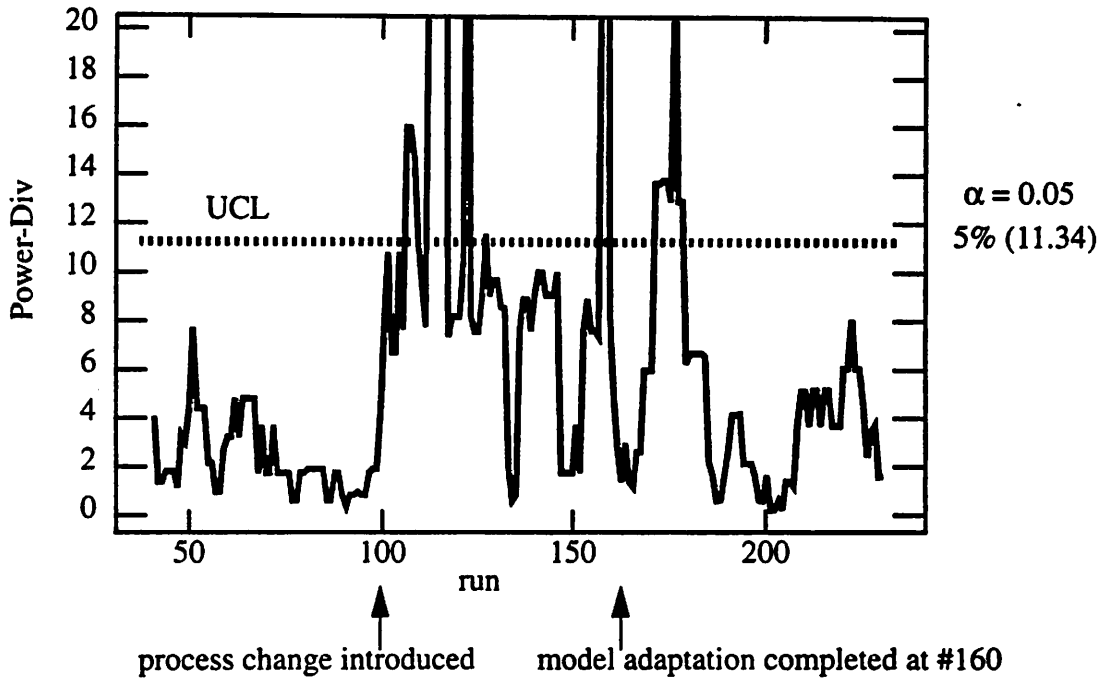


Figure 3.9 A Model Adaptation Example with Locked Process Settings

A second model adaptation example is shown in Figure 3.10. This example is similar to the first example, except the process is *unlocked:* the input variables are allowed to vary 50% randomly but uniformly within the center half of the original experimental range.

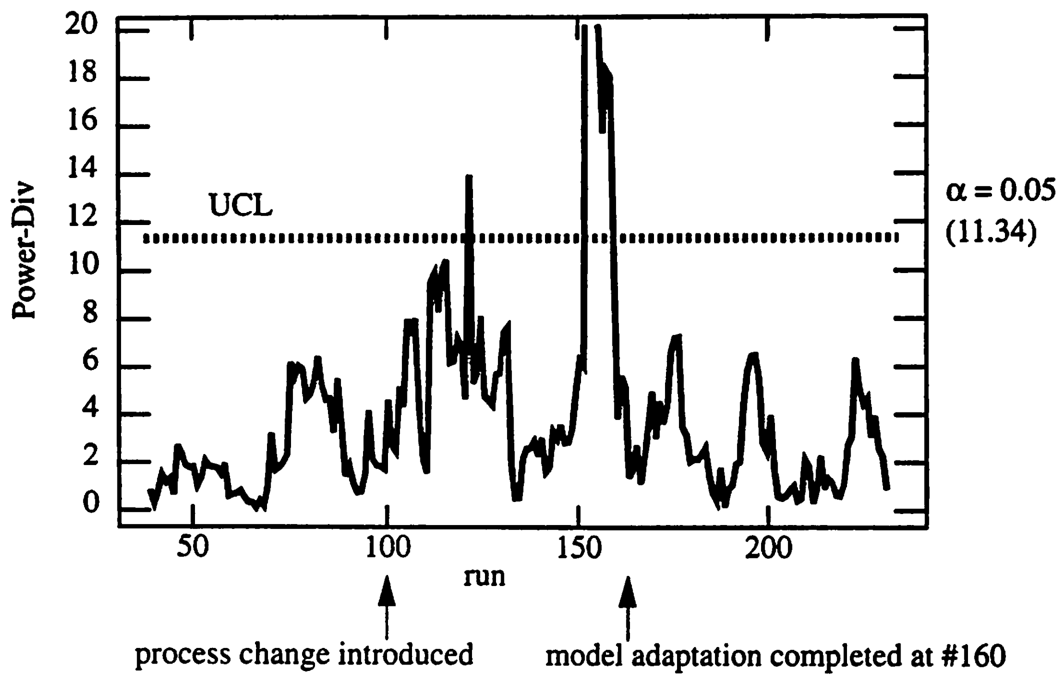


Figure 3.10 A Model Adaptation Example with Unlocked Process Settings

3.4 Summary

A systematic methodology for model-based SPC was presented that takes advantage of qualitative process observations. The IRLS meth[illegible]les is consistently c models can be changed once, or continuously. Updated models are locally accurate. Since the adaptation is based on new probability estimates, it takes 30-50 observations after the change to estimate new frequencies of occurrence. Given these statistical tests, model updates can be easily automated. Even though its implementation is straightforward, the methodology offers a novel approach to characterize and use important, yet informal qualitative knowledge about a process.

3.5 References

[3.1] D.W. Hosmer and L. Lemeshow, *Applied Logistic Regression*, New York: John Wiley, 1989.

[3.2] D.F. Morrison, *Multivariate Statistical Methods*, 3rd edition, pp 87-89, New York: McGraw-Hill, 1990.

[3.3] D.C. Montgomery, *Introduction to Statistical Quality Control*, 2nd Edition, New York, John Wiley & Sons, 1991.

[3.4] S. Katz and N.L. Johnson, *Encyclopedia of Statistical Sciences*, vol. 1, pp. 439-442, New York, John Wiley & Sons, 1982.

[3.5] SAS User's Guide, SAS Institute Inc., SAS Campus Drive, Cary, NC 27513.

Chapter 4 A Self-Learning Fuzzy Inference System

The methods in the previous two chapters about categorical data analysis are direct generalizations of conventional statistical analysis for continuous variables. Even though the approach is useful and informative, its usage is limited since the adaptive model needs large amounts of data to be effective. Also, knowledge obtained from one process is difficult to transfer to another process. In this chapter, we will introduce an approach based on fuzzy logic inference, to create systems appropriate for qualitative features. In Chapter 5, we will combine the formality of the statistical approach to the flexibility of a fuzzy inference system.

Fuzzy logic has been applied successfully in many fields where qualitative knowledge and data are involved. In this chapter, we explore a fuzzy logic application on process modeling. Specifically, a prototype qualitative model based on experimental data is created to predict the average grain size of polysilicon deposition films through a fuzzy inference system.

The overall goal of this work is to develop a software system that uses fuzzy models for the inference of qualitative aspects of a process (see Figure 4.1). Within this system, models are developed from experimental data to describe qualitative process attributes, which are usually understood by an experienced process engineer, but ignored by most quantitative approaches. A low pressure chemical vapor deposition (LPCVD) furnace for polysilicon deposition has been selected as a test vehicle for this investigation [4.1]. Prototype models can be developed to predict process responses such as grain size, surface

roughness, grain orientation and step coverage of a deposited film, based on recipe settings, i.e., deposition/annealing temperature, dopant flow rate, silane flow rate, pressure, and deposition time, etc. The combined quantitative-qualitative models can be used for equipment and process simulation, recipe generation, diagnosis, and control. Similar qualitative models may also be developed for other equipment and processes.

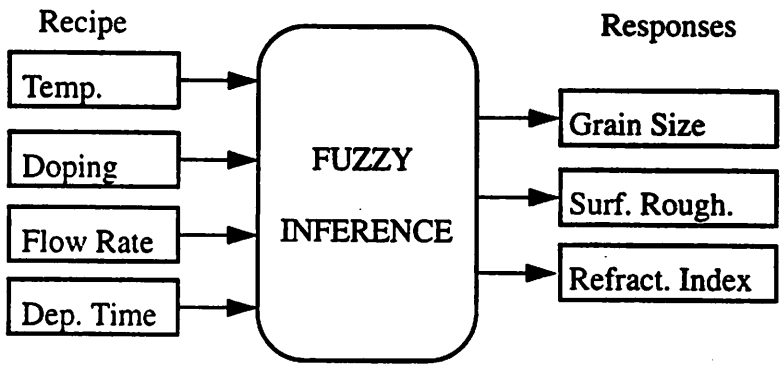


Figure 4.1 A Fuzzy Inference System for LPCVD Process

4.1 Background Knowledge

4.1.1 Low pressure chemical vapor deposition process

Figure 4.2 depicts the Tylan horizontal glass tube reactor used for polysilicon LPCVD [4.2]. The wafers are stacked perpendicularly to the longitudinal axis of the tube, and are placed at 1.2 *cm* intervals. Aluminum cantilever rods support the boats, which carry the wafers. Silane (SiH_4), mixed with an inert carrier gas (N_2), is injected at the lower front end of the tube, and is pumped out from the back end. Three main heating coils are placed along the tube to maintain the proper operating temperature. A heat baffle in the front portion of the furnace reduces radiative heat loss and also serves to facilitate both temperature and gas velocity control in the middle and back sections of the tube.

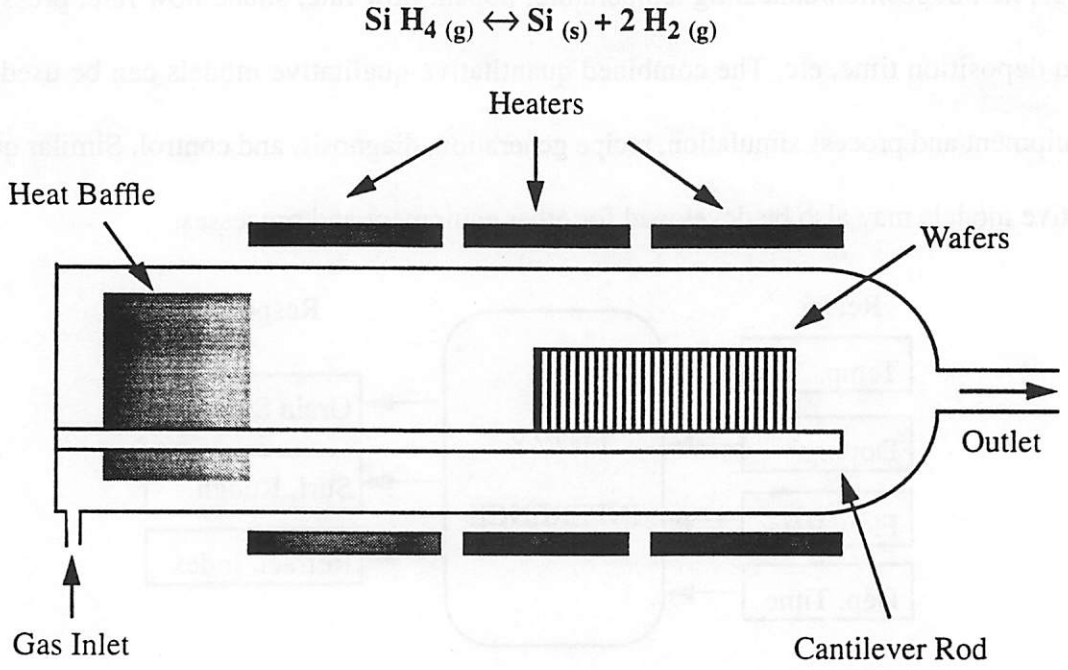


Figure 4.2 Deposition Furnace (*Courtesy*: Sherry F. Lee [4.2])

There are many qualitative properties that cannot be modeled numerically, such as grain size, surface smoothness, etc., even though extensive research has been done on those aspects and much knowledge has been accumulated. However, these qualitative characteristics of the polysilicon film are very important to semiconductor devices. A well-built control system must capture, at least qualitatively, human knowledge about relationships between the LPCVD process settings and the process responses in order to make the process more controllable and enhance the quality of the devices. As shown in this chapter, an inference system based on fuzzy logic would be a good candidate to capture qualitative human knowledge.

4.1.2 Grain size prediction

Among other quantitative and qualitative features of an LPCVD deposited polysilicon film, the average grain size (s) is one of the important quantities that characterize the structure and property of the film and that will affect the semiconductor device performance [4.3]. The grain size can be measured in a number of different ways. Figure 4.3 and Figure 4.4 show how a polysilicon film structure responds to the different process setting of deposition temperature (T_d), annealing temperature (T_a), and doping profile [4.3].

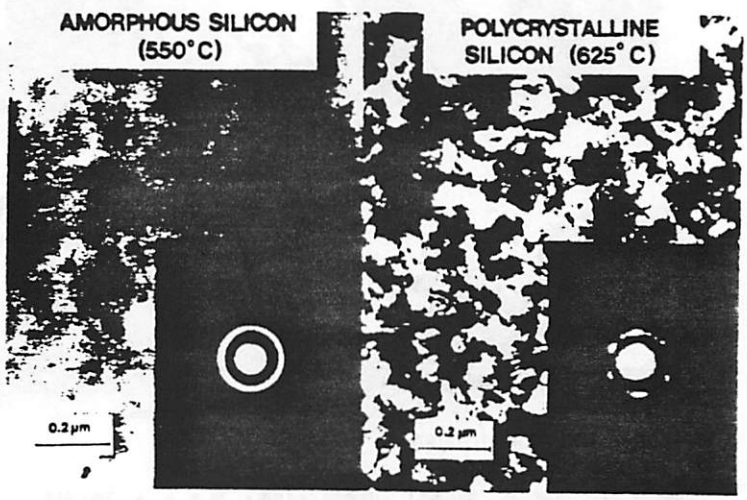


Figure 4.3 Grain Size (Structure) vs. T_d (*Courtesy*: T. Kamins [4.3])

Figure 4.3 shows a transmission electron micrograph of 0.6 μm thick polysilicon films deposited onto S_iO_2 at 550°C and 625°C in an LPCVD reactor. The deposition temperature T_d does have an effect on the grain structure. The structure of a polysilicon film deposited at 625°C has an average grain size of about 70 *nm*.

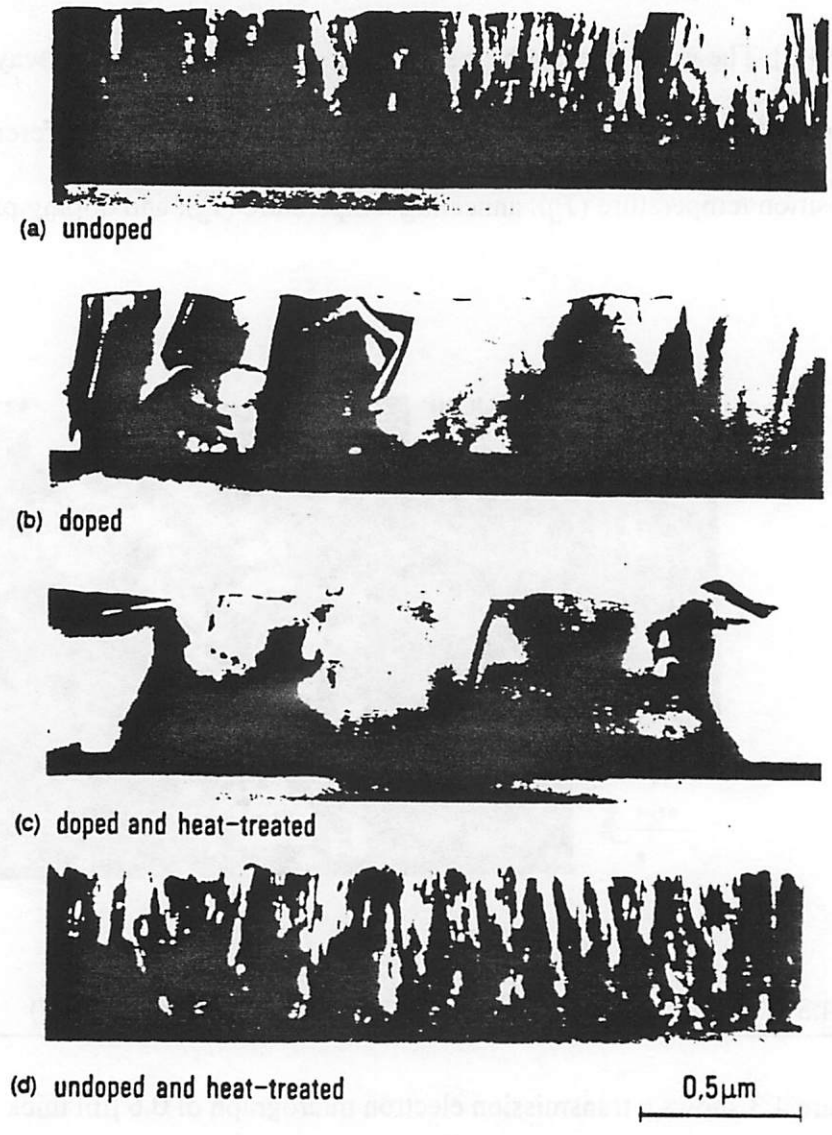


Figure 4.4 Grain Size (Structure) vs. T_a and Doping Profile (*Courtesy*: T. Kamins [4.3])

Figure 4.4 gives another example of how grain structure (and grain size) could be

affected by different recipe settings during deposition. It shows cross section transmission electron micrographs of 0.6 μm thick polysilicon films deposited in an LPCVD reactor at T_d = 625°C. The initial columnar structure of undoped polysilicon films (a) changes little when the undoped films are annealed (d), but phosphorus doping increases the grain size (b), which is further enlarged by additional annealing of the doped films (c). Other experiments show that for phosphorus doped polysilicon films, the grain size of the polysilicon structure increases when T_d increases, however, after high temperature (T_a) annealing, the films deposited with lower T_d would have larger average grain size than those deposited with higher T_d [4.4].

To date, there are no generally accepted theories that explain the complicated relations between deposition/annealing temperatures and resulting grain size (s). The expert knowledge on this field is very qualitative and sometimes contradictory. The proposed model in this chapter manages to capture the empirical qualitative knowledge into a computer software system that would eventually help process engineers better understand the subject. Since this system is based on fuzzy logic theory, a brief introduction to fuzzy logic and a fuzzy inference decision-making system is given in the following section.

4.2 An Introduction to Fuzzy Logic

The concept of Fuzzy Logic was first introduced by Professor Lotfi A. Zadeh [4.5] of the University of California at Berkeley, in June 1965. In the last few years, the subject has flourished and applications of this theory can now be found in many disciplines.

4.2.1 Fuzzy sets and membership functions

Central to Fuzzy Logic is the concept of the fuzzy set [4.6]. The fuzzy set theory is in many ways a generalization of the classical set theory. A *classical* (crisp) set A is normally defined as a collection of elements or objects x (belong to a super-set X) which satisfy certain conditions specifying A. Each element $x \in X$ can either belong to or not belong to the set A, where $A \subseteq X$. To generalize this definition, we can introduce a membership function μ (on X) for each element x in order to quantify its *belongness* to the crisp set A, i.e. $\mu(x)=1$ if $x \in A$ and $\mu(x)=0$ if $x \notin A$. If we allow this membership function to be continuous, we can define a *fuzzy* set B as a collection of elements $x \in X$ with membership function $\mu(x)$, where $\mu(x)$ can be any real number between 0 and 1:

$$B = \{ (x, \mu_B(x)) \,|\, x \in X \} \tag{4-1}$$

4.2.2 Fuzzy logic and rules

In Boolean logic, the most basic logic operations we need to consider are "PASS", "COMPLEMENTARY", "AND" and "OR". The rules of those operations and their "fuzzy" generalizations are expressed as shown in Table 4.1:

Table 4.1 Boolean Logic Rules

Name	Crisp Rule	Equivalent Fuzzy Membership Value
PASS	IF $x \in A$, THEN $z \in Z$	IF $\mu_A(x)=1$, THEN $\mu_Z(z)=1$
COMPL.	IF $x \notin A$, THEN $z \in Z$	IF $\mu_A(x)=0$, THEN $\mu_Z(z)=1$
OR	IF $x \in A \vee y \in B$, THEN $z \in Z$	IF $\mu_A(x)=1$ OR $\mu_B(y)=1$, THEN $\mu_Z(z)=1$
AND	IF $x \in A \wedge y \in B$, THEN $z \in Z$	IF $\mu_A(x)=1$ AND $\mu_B(y)=1$, THEN $\mu_Z(z)=1$

where "1" can be defined as "true" and "0" as "false".

Thus fuzzy logic can be regarded as a generalization of the classical Boolean logic by allowing the "membership function" $\mu(x)$ to be any number between 0 and 1. Equivalently, Boolean logic is a special case of fuzzy logic. A set of equivalent fuzzy logic rules ("fuzzy rules") can be defined for the above four basic logic operations as shown in Table 4.2:

Table 4.2 Fuzzy Logic Rules

Name	Rule	Membership Value
PASS	IF $(x, \mu_A(x)) \in A$, THEN $(z, \mu_Z(z)) \in Z$	$\mu_Z(z) = \mu_A(x)$
COMPL.	IF $(x, \mu_A(x)) \in A$, THEN $(z, \mu_Z(z)) \in Z$	$\mu_Z(z) = 1-\mu_A(x)$
OR	IF $(x, \mu_A(x)) \in A$ OR $(x, \mu_B(x)) \in B$, THEN $(z, \mu_Z(z)) \in Z$	$\mu_Z(z) = \max(\mu_A(x), \mu_B(x))$
AND	IF $(x, \mu_A(x)) \in A$ AND $(x, \mu_B(x)) \in B$, THEN $(z, \mu_Z(z)) \in Z$	$\mu_Z(z) = \min(\mu_A(x), \mu_B(x))$

where all of the values of the membership functions are between 0 and 1. Note that the above definition given for the "OR and "AND" are not unique. Other definitions are also in use in the literature [4.6].

4.2.3 The concept of the linguistic variable

A linguistic variable has a name, and a set of *linguistic* values [4.6]. Each of these linguistic values is associated with a value of a membership function. For example: a person's *age A* can be a linguistic variable taking linguistic values from the set {*juvenile, young, middle-aged, old, very old*}. The relationship between the numerical value of the age and the corresponding linguistic values can be captured by membership functions as shown in Figure 4.5.

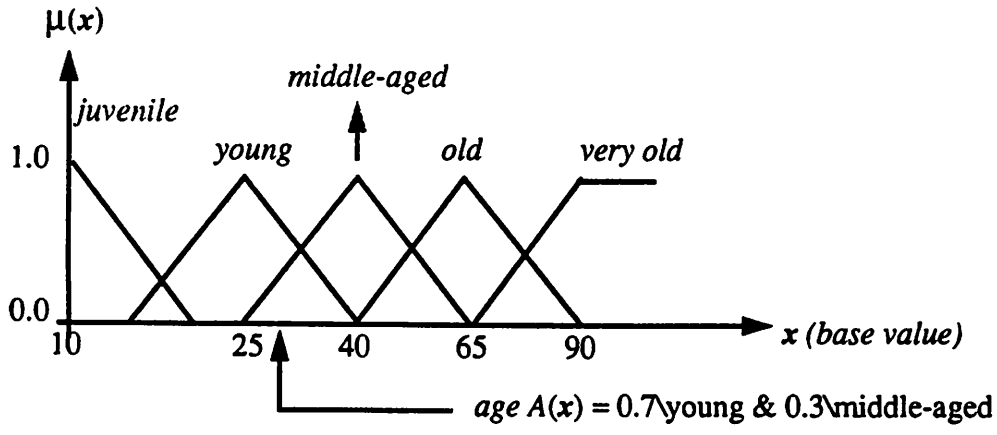


Figure 4.5 Membership Functions of the Various Age Groups

Consider that if x=30, the value of the linguistic variable "age" is then $A(x)$ = 0.7\young & 0.3\middle-aged, which means that for x=30, the linguistic variable A has the value "young" with a membership of 0.7 and the value "middle-aged" with a membership of 0.2.

Once we have captured information in terms of linguistic variables, we can then use fuzzy logic to describe the relationship among such variables. For instance, a rule may exist which states: IF A is young THEN P is energetic, where P is another linguistic variable that stands for a person's physical condition and takes values from the set {*inactive, energetic*}. Using fuzzy logic terminology, this rule can be expressed as:

$$\mu_{\text{energetic}}(P) = \mu_{\text{young}}(A).$$

The concept is depicted in Figure 4.6.

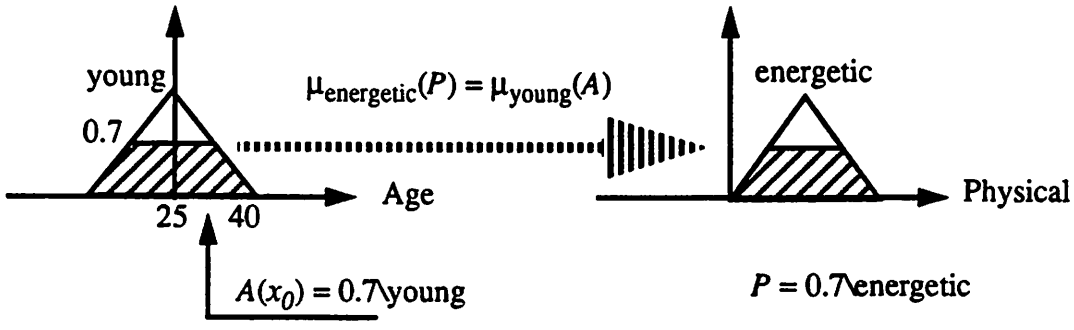


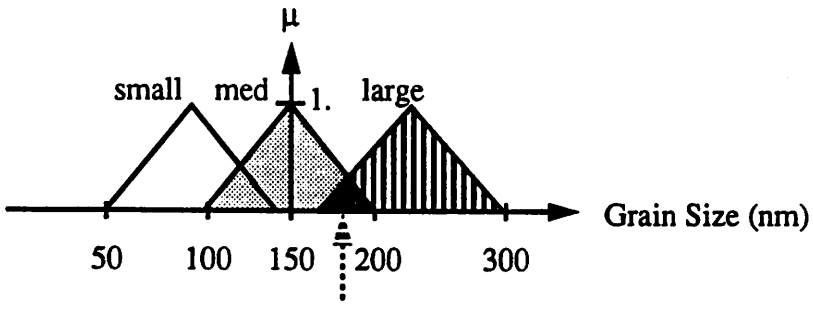
Figure 4.6 Example of a Fuzzy Logic Inference Rule

This is actually a simple case of *fuzzy inference* with only one input (A) and one output (P). More general cases of fuzzy inference will be introduced in the next section.

4.2.4 Fuzzy inference

Fuzzy inference is an "approximate reasoning" technique, based on fuzzy logic rules, linguistic variables, and their membership functions. The concept will be illustrated through examples that use the model of a polysilicon LPCVD process.

One example is determining the grain size (s) given the deposition temperature (T_d) and annealing temperature (T_a) of a process. These three parameters (T_d, T_a, and s) should be *fuzzified* to linguistic variables by representing them through fuzzy sets and membership functions. For example, in LPCVD of polysilicon, a grain size of 180nm is fuzzified using the membership functions depicted in Figure 4.7:



Grain Size = {(small, 0), (medium, 0.3), (large, 0.2)}

Figure 4.7 Grain Size

The following fuzzy rules for annealed polysilicon films were deduced from expert knowledge:

1. IF the deposition temperature is low, THEN the grain size is large.

2. IF the annealing temperature is high, THEN the grain size is large.

These rules can be summarized in a rule table as shown in Table 4.3:

Table 4.3 Fuzzy Rule Table

Grain Size	T_a low	T_a med	T_a high
T_d low			large
T_d med		med	
T_d high	small		

Now consider a set of input values pairs T_d, T_a. The fuzzified inputs can be represented by linguistic values through their membership values:

$T_d = 0.8\backslash\text{low} \;\&\; 0.25\backslash\text{med};$

$$T_a = 0.45\backslash\text{med} \;\&\; 0.5\backslash\text{high};$$

By applying the fuzzy rules of Table 4.3[1], the output membership values are:

$$w_1(T_d = \text{low}, T_a = \text{high}) = \min(0.8, 0.5) = 0.5;$$

$$w_2(T_d = \text{med}, T_a = \text{med}) = \min(0.25, 0.45) = 0.25.$$

Thus the output grain size s is:

$$s = 0.25\backslash\text{med} \;\&\; 0.5\backslash\text{large}.$$

Finally, a "defuzzification" technique should be applied to obtain a numerical value of an output linguistic variable after fuzzy inference. There are many ways to defuzzify an output. A simple way is to do a "weighted average":

$$s = \frac{w_1 \cdot A + w_2 \cdot B}{w_1 + w_2} \tag{4-2}$$

where A and B are corresponding numerical values for s = large and s = med.

1. There are many ways other than the "minimum" operation to obtain the weights w_1 and w_2 from the "AND" rules for multiple inputs. An alternative is to use the product of membership values of each individual input (see Section 4.3).

The entire inference algorithm is illustrated in Figure 4.8.

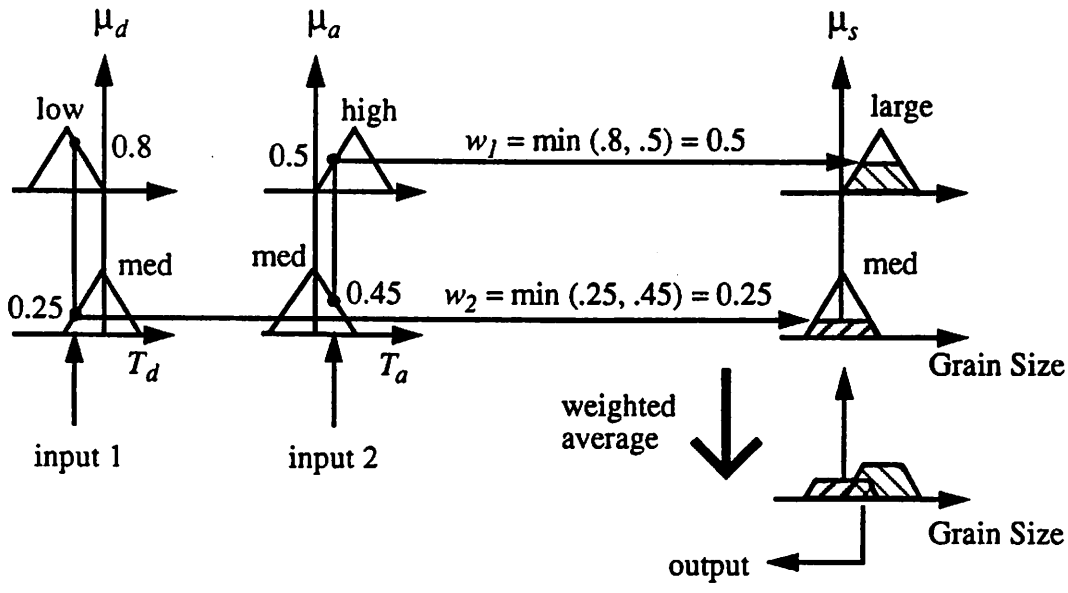


Figure 4.8 Fuzzy Inference

In a more general situation, different rule uses may lead to "conflicting" results, i.e., there may be two or more different weights for one specific output value. The proper inference procedure is to sum up the weights for the same output (linguistic) value and then apply the above defuzzifying algorithm to obtain the numerical output value.

In the next section, a more generic grain size prediction model based on input values of (T_d,T_a) will be presented. The rules for the inference system will be derived directly from the existing experimental data, or "training data".

4.3 Self-Learning LPCVD Models

4.3.1 Representation and inference of a general fuzzy system

A fuzzy inference system with 2 independent inputs $\{x_1, x_2\}$ and one output y is considered in this section. This system may be directly generalized to an n-input system [4.7].

The inference rules of such a 2-input fuzzy system are expressed as:

- Rule-$\{i,j\}$ IF x_1 is A_{1i} and x_2 is A_{2j} THEN y is w_{ij}.

where $\{i,j\}$ (i=0,1,...,m, j=0,1,...,n) are rule numbers, A_{1i} and A_{2j} are membership functions of the antecedent part, and w_{ij} is a real number of the consequent part of the output value of y.

Without loss of generality, the membership functions A_{1i} are expressed by triangles as shown in Figure 4.9:

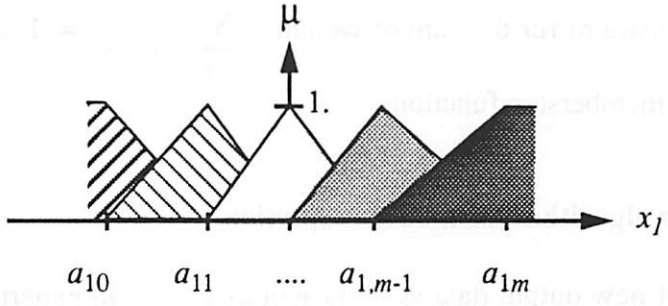


Figure 4.9 Membership Functions for x_1

The expressions that correspond to the triangle membership functions are:

$$A_{1i} = \begin{cases} \dfrac{x_1 - a_{1,i-1}}{a_{1i} - a_{1,i-1}}, & \text{when } a_{1,i-1} < x_1 \le a_{1i}; \\ \dfrac{a_{1,i+1} - x_1}{a_{1,i+1} - a_{1i}}, & \text{when } a_{1i} < x_1 \le a_{1,i+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-3}$$

The expressions for A_{2j} can be obtained simply by changing the subscript "1" to "2".

Here, when applying the Rule-$\{i,j\}$, we use a different version of the "AND" rule applied in Section 4.2. Namely, we use the product of A_{1i} and A_{2j}, instead of their minimum value, in order to obtain the weight for output value w_{ij}. The defuzzified predicted output $\hat{y}$ thus can be derived by weighted average:

$$\hat{y} = \sum_{i,j} A_{1i} \cdot A_{2j} \cdot w_{ij} \tag{4-4}$$

In this case, the constraint for the sum of weights $\sum_{i,j} A_{1i} \cdot A_{2j} = 1$ is the result of the above definition of membership functions.

4.3.2 A self-tuning algorithm for model adaptation

Assume a set of new output data y_k^{new} is acquired from an experimental data-base, that corresponds to input data $\{x_{1k}, x_{2k}\}$ (k=1, ..., q). The fuzzy inference system will learn and fit these data by adjusting the parameters $\{a_{1i}, a_{2j}, w_{ij}, i=0, 1, ..., m, j=0, 1, ..., n\}$. Namely, the system parameters $\{a_{1i}, a_{2j}, w_{ij}\}$ will be decided based on a data set $\{x_{1k}, x_{2k}, y_k^{new}\}$.

First, a cost function is first defined:

$$E = \frac{1}{2}\sum_{k=1}^{q}\left(\hat{y}_k - y_k^{\text{new}}\right)^2 = \frac{1}{2}\sum_{k=1}^{q}\left(\sum_{i,j} A_{1i}\cdot A_{2j}\cdot w_{ij} - y_k^{\text{new}}\right)^2 \tag{4-5}$$

where A_{1i} and A_{2j} are also functions of x_{1k} and x_{2k} as shown in their definitions above. To solve the ensuing optimization problem, the derivatives of E need to be calculated, which are shown as:

$$\frac{\partial E}{\partial w_{ij}} = \sum_{k=1}^{q}\left(\hat{y}_k - y_k^{\text{new}}\right)\cdot A_{1i}\cdot A_{2j} \tag{4-6}$$

and,

$$\frac{\partial E}{\partial a_{1i}} = \sum_{j=0}^{n}\sum_{k=1}^{q}\left(\hat{y}_k - y_k^{\text{new}}\right)\cdot\left(\sum_{l=i-1}^{i+1} u_l\cdot w_{lj}\right) \tag{4-7}$$

$$\frac{\partial E}{\partial a_{2j}} = \sum_{i=0}^{m}\sum_{k=1}^{q}\left(\hat{y}_k - y_k^{\text{new}}\right)\cdot\left(\sum_{l=j-1}^{j+1} v_l\cdot w_{lj}\right) \tag{4-8}$$

where,

$$u_i = \begin{cases} \dfrac{-A_{1i} \cdot A_{2j}}{a_{1i} - a_{1,i-1}}, & \text{when } a_{1,i-1} < x_1 \le a_{1i}; \\ \dfrac{A_{1i} \cdot A_{2j}}{a_{1,i+1} - a_{1i}}, & \text{when } a_{1i} < x_1 \le a_{1,i+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-9}$$

$$u_{i-1} = \begin{cases} \dfrac{(1 - A_{1,i-1}) \cdot A_{2j}}{a_{1i} - a_{1,i-1}}, & \text{when } a_{1,i-1} < x_1 \le a_{1i}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-10}$$

$$u_{i+1} = \begin{cases} \dfrac{(A_{1,i+1} - 1) \cdot A_{2j}}{a_{1,i+1} - a_{1i}}, & \text{when } a_{1i} < x_1 \le a_{1,i+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-11}$$

and similarly,

$$v_j = \begin{cases} \dfrac{-A_{1i} \cdot A_{2j}}{a_{2j} - a_{2,j-1}}, & \text{when } a_{2,j-1} < x_2 \le a_{2j}; \\ \dfrac{A_{1i} \cdot A_{2j}}{a_{2,j+1} - a_{2j}}, & \text{when } a_{2j} < x_2 \le a_{2,j+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-12}$$

$$v_{j-1} = \begin{cases} \dfrac{A_{1i} \cdot (1 - A_{2,j-1})}{a_{2j} - a_{2,j-1}}, & \text{when } a_{2,j-1} < x_2 \le a_{2j}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-13}$$

$$v_{j+1} = \begin{cases} \dfrac{A_{1i} \cdot (A_{2,j+1} - 1)}{a_{2,j+1} - a_{2j}}, & \text{when } a_{2j} < x_2 \le a_{2,j+1}; \\ 0, & \text{otherwise.} \end{cases} \tag{4-14}$$

The constraints of those parameters are:

$$\begin{cases} a_{10} \le a_{11} \le \ldots\ldots \le a_{1m}, \\ \\ a_{20} \le a_{21} \le \ldots\ldots \le a_{2n}. \end{cases} \tag{4-15}$$

There are many optimization techniques available to solve this problem. The BCAM system's non-linear variable metric constraint optimization package Hanpal [4.8][4.9] is used to obtain the $(m + 1) \times (n + 1) + q$ parameters $\{a_{1i}, a_{2j}, w_{ij}\}$.

4.3.3 Simulation results for grain size prediction

The 12 data points in Figure 4.10 are past experimental result of X-ray-measured average grain size of phosphorus-doped LPCVD polysilicon films. The first 9 representing date points are used as training data to derive 9=3x3 rules for the inference system (i.e. m=n=2, see previous section). By applying the fuzzy inference algorithm illustrated in Section 4.3.1, the system interpolates all the data points for a 2-dimensional region of input space (T_d, T_a), as shown by the curves in Figure 4.10(a).

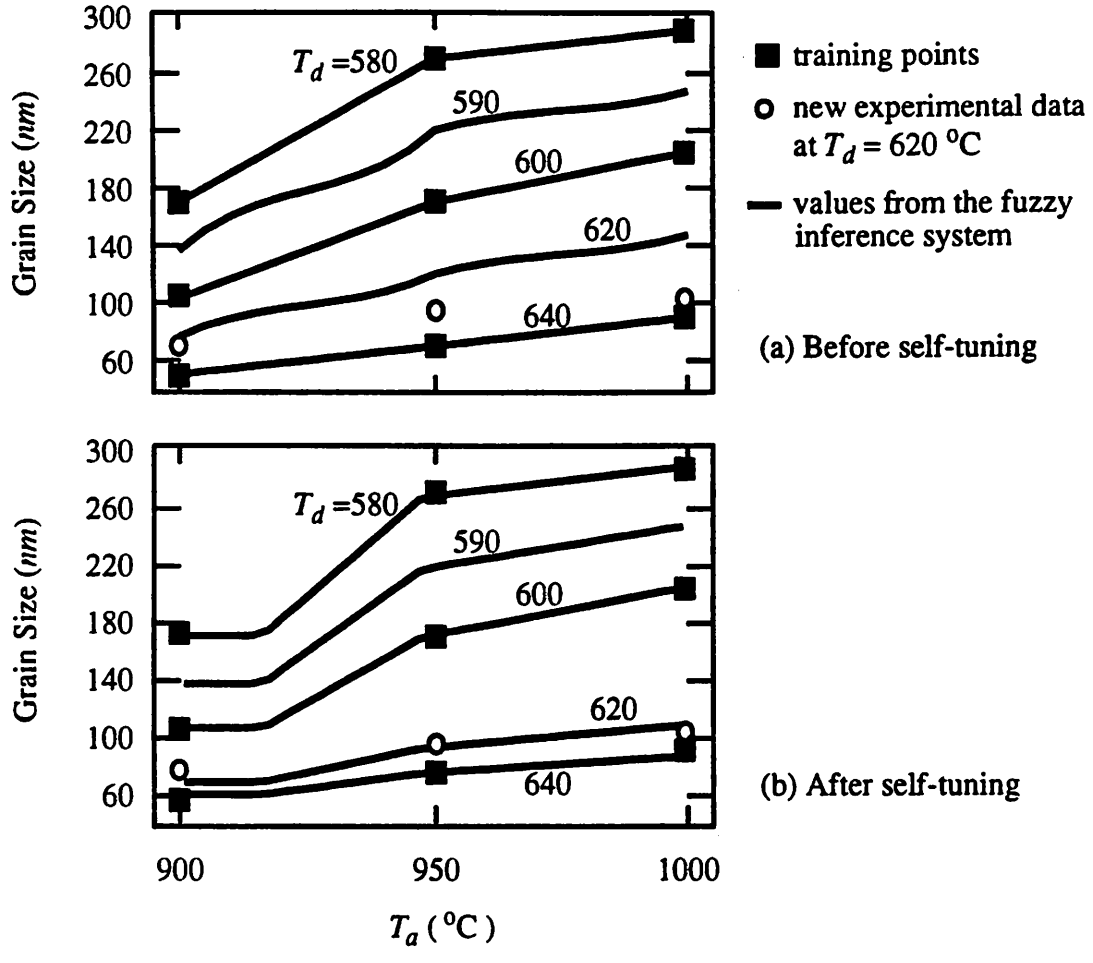


Figure 4.10 X-ray Grain Size versus Annealing and Deposition Temperatures (LPCVD)

The remaining 3 data points are newly introduced data and the system tries to best fit all the 12 data points by optimizing the cost function as defined in Eq. (4-5). Note that while the number of rules does not change, the membership functions are modified by changing the parameters $\{a_{1i}, a_{2j}\}$ and the corresponding output parameters w_{ij} ($i,j = 0,1,2$ in this case). The total number of optimized parameters is 15 (= m+n+2+(m+1)x(n+1)). The result is shown in Figure 4.10(b), where the curves are now fitting all of the 12 points. For this example, the cost function had the initial value of 1.44×10^5 and was reduced to 1.18×10^4 in 28 iterations, as shown in Figure 4.11.

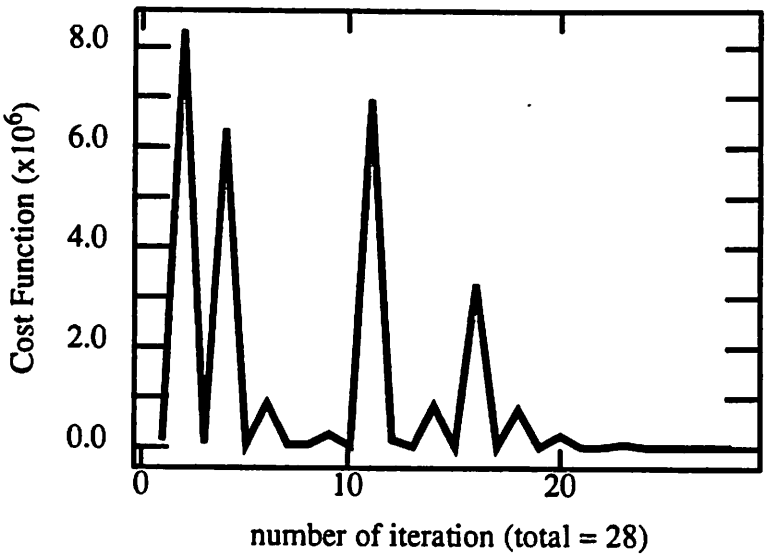


Figure 4.11 Cost Function versus the Number of Iteration

Finally, the membership functions before and after the optimization process for representing the two input linguistic variables (T_d, T_a) are shown in Figure 4.12. The numerical rules are shown below:

- 9 training data points:

$$\begin{cases} a_1 = [a_{1i}] = \begin{bmatrix} 580 \\ 600 \\ 640 \end{bmatrix}, \qquad a_2 = [a_{2j}] = \begin{bmatrix} 900 \\ 950 \\ 1000 \end{bmatrix}, \\ w = [w_{ij}] = \begin{bmatrix} 170 & 270 & 290 \\ 103 & 170 & 205 \\ 50 & 70 & 90 \end{bmatrix}, \qquad \text{where } i, j = 1, 2, 3. \end{cases} \tag{4-16}$$

- 3 new data points:

$$\begin{cases} \left(x_1, x_2, y^{new}\right) = \left([x_{1k}], [x_{2k}], [y_k^{new}] \right) \\ \qquad\qquad = \begin{bmatrix} 620 & 900 & 75 \\ 620 & 950 & 95 \\ 620 & 1000 & 100 \end{bmatrix} \quad \cdot \quad \text{where } k=1,2,3. \end{cases} \tag{4-17}$$

where the unit for temperature parameters a_{1i}, a_{2j} and x_{1k}, x_{2k} is ^{o}C, and the unit for the grain size parameter w_{ij} and the desirable output data points y_k^{new} is *nm*.

After the solution of the optimization problem, the 15 new parameters are:

$$\begin{cases} a_1 = \begin{bmatrix} 579.8 \\ 598.0 \\ 624.5 \end{bmatrix}, \qquad a_2 = \begin{bmatrix} 915.4 \\ 947.4 \\ 999.2 \end{bmatrix}, \\ w = \begin{bmatrix} 170.8 & 270.0 & 291.0 \\ 108.5 & 176.8 & 213.5 \\ 57.4 & 72.5 & 84.6 \end{bmatrix}, \qquad \text{where } i,j = 1,2,3. \end{cases} \tag{4-18}$$

as plotted in the lower parts of Figure 4.12(a) and Figure 4.12(b).

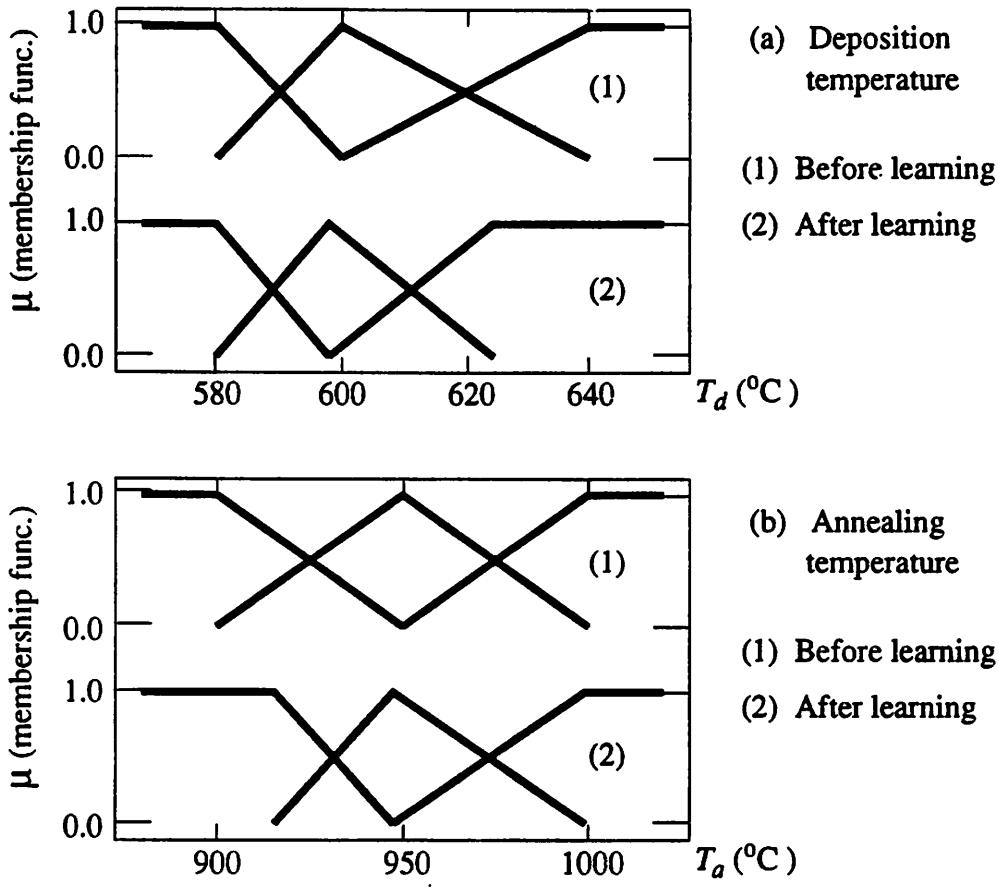


Figure 4.12 Membership Functions Before and After Learning

Our results show that the fuzzy logic based inference system can represent qualitative knowledge well. Also, it can adaptively adjust the system parameters to accommodate new data.

The combined quantitative and qualitative models can be used for LPCVD recipe generation and process control. The knowledge base may be further modified and enhanced after taking into account actual process responses. (See Figure 4.13).

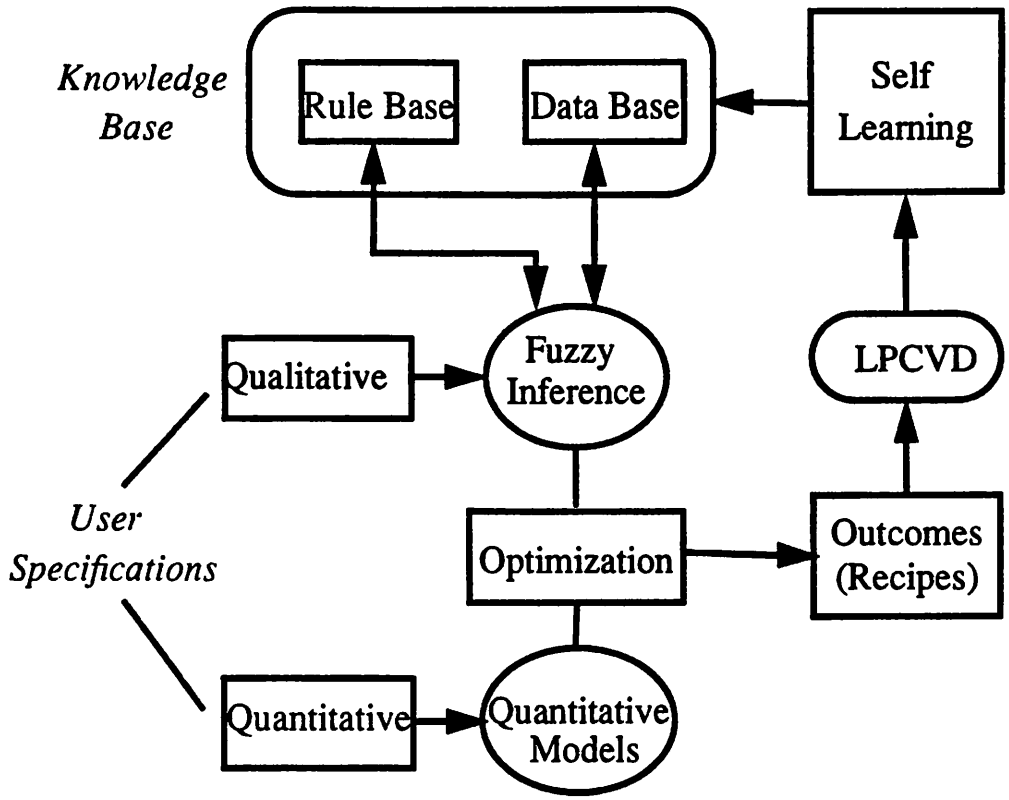


Figure 4.13 Architecture of an LPCVD Equipment Model

4.4 Summary

Due to the large number of process variables and their complex interactions in a semiconductor manufacturing environment, the pertinent expert knowledge is mostly qualitative and incomplete. To build a computer-aided integrated system for future intelligent manufacturing applications, fuzzy logic based qualitative models can be used to capture qualitative human knowledge in a computer software simulation system. This fuzzy inference system can be self-learning to accommodate updated expert knowledge.

According to fuzzy logic methodology, the process parameters are first fuzzified into linguistic variables before the inference rules are applied. These linguistic values of the process parameters are then mapped to appropriate linguistic values of the output parameters. The mapping is implemented as a series of fuzzy set operations. The proper input-output relationships are captured by means of designing the appropriate membership functions and inference rules through the existing training data. The output linguistic values obtained from the fuzzy inference system are finally defuzzified into conventional numerical values.

When the new process data is obtained, the system automatically adjusts the membership functions and the corresponding fuzzy rule parameters to best fit the new data. This self-learning algorithm has great potential in applications on real-time process control in a computer-aided manufacturing environment.

Often, however, experimental data is scarce and noisy. In the next chapter, we will discuss the application of formal statistics in dealing with such data before we use them for the creation of a fuzzy inference system.

4.5 References

[4.1] K.K. Lin and C.J. Spanos, "Statistical Equipment Modeling for VLSI Manufacturing: An Application to LPCVD Reactors," *IEEE Transactions on Semiconductor Manufacturing*, Vol.3, No.4, pp. 216-229 (November 1990).

[4.2] S.F. Lee, "A Three-Dimensional Physically-Based LPCVD Model," *M.S.E.E.*

Thesis, EECS Dept., U.C.Berkeley (1992).

[4.3] T. Kamins, *Polycrystalline Silicon for Integrated Circuit Applications*, Kluwer Academic Publishers (1988).

[4.4] G. Herbage, L. Krausbauer, E.F. Steigmeier, A.E. Widmer, H.F. Kappert, and G. Neugebauer, "LPCVD polycrystalline silicon: Growth and physical properties of *in-situ* phosphorus doped and undoped films," *RCA Review*, Vol.**44**, 287-312 (June 1983).

[4.5] L.A. Zadeh, "Fuzzy Sets," *Information and Control*, Vol.**8**, 338-353 (1965).

[4.6] H.-J. Zimmermann, *Fuzzy Set Theory - and its Applications*, Kluwer Academic Publishers (1991).

[4.7] H. Nomura, I. Hayashi and N. Wakami, "A Learning Method of Fuzzy Inference Rules by Descent Method," proceedings of *IEEE International Conference on Fuzzy Systems*, pp.203-210, San Diego (March 8-12, 1992).

[4.8] C.J. Spanos, "Statistical Parameter Extraction for IC Process Characterization," *Ph.D dissertation*, Dept. of Electrical and Computer Engineering, Carnegie-Mellon University (May 1985). (The optimization program package Hanpal is available at BCAM workstation cluster: spanos@radon.berkeley.edu:prometheus/).

[4.9] M.J.D. Powell, "A Fast Algorithm for Nonlinearly Constrained Optimization Calculation," *Proceedings of 1977 Dundee Conference on Numerical Analysis* (June 1977).

Chapter 5 Using Categorical Analysis To Design A Fuzzy Inference System

In this chapter we propose a systematic method for designing a fuzzy inference system through statistical data pre-processing. This approach is appropriate in modeling the qualitative aspects of a semiconductor manufacturing process, when extensive training data are often limited or difficult to collect due to the high cost of conducting experiments. In view of the limited number of data sets from a designed experiment, this system employs the proper statistical analysis in order to highlight the significant effects. Simple fuzzy inference rules are then created to capture input-output relationships and to initialize the corresponding membership functions. The output process variable can be continuous or categorical, and the fuzzy system can be further tuned to accommodate additional experimental data when they become available.

As shown in Chapter 4, in a fuzzy inference modeling system the process parameters become linguistic variables and the qualitative relationships between input and output parameters are represented by fuzzy inference rules. Such a system can be adapted by *tuning* the fuzzy membership functions to accommodate newly acquired experimental data. However, the initial design of such a system requires the determination of the number of rules as well as the number and initial form of the membership functions. This problem is complicated by the large number of input factors, and by the high cost of acquiring extensive experimental data in a semiconductor manufacturing environment.

To solve this problem, a statistical data pre-processing method based on linear and

logistic regression analysis can be used (see Chapter 2). Starting from the experimental data obtained from statistically designed experiments, statistical analysis can screen out insignificant input variables for a specific output response, thus reducing the complexity of the problem. At the same time, the rules and initial membership functions of the inference system can be extracted from the results of the statistical analysis. The system can be subsequently refined and adapted to newly acquired experimental data by further tuning its membership functions.

5.1 Knowledge Extraction from Designed Experiments

As shown in Section 2.4, the following table gives an example of data from a designed experiment [5.1] for a dry develop process[1].

Table 5.1 A Dry Develop Process

Power (w)	Pressure (p)	Oxygen Flow (O_2)	*roughness* {smooth, fair, rough, roughest}	*mouse bites* {good, fair, poor, worst}
1	-1	1	roughest	poor
-1	0	-1	fair	fair
0	0	0	fair	poor
...	...	...	...	...
-1	-1	0	smooth	good
-1	1	-1	fair	fair
0	-1	-1	rough	worst
0	0	0	rough	fair

Starting from the data table obtained from the 3-level factorial design shown, linear regression analysis is used for *interval* variables (e.g. the average line width *lw*) and logistic regression analysis is applied for *categorical* variables (e.g. *mouse bites*). Only the sta-

1. Input values have been normalized.

tistically significant input(s) are then considered for building the fuzzy inference system for each output variable. To simplify the explanation of such a fuzzy system building process, we consider first one input variable for each output response. In this way, we will illustrate our fuzzy system design method through the examples of modeling a continuous output variable (e.g. line-width *lw*) vs. an input variable (e.g. power *w*), and a categorical output variable (e.g. *mouse bites*) vs. an input variable (e.g. pressure *p*). Multi-input situations will be discussed afterwards.

Assume that the linear fitting curve for *lw* vs. *w* from the linear regression analysis is as shown in Figure 5.1,

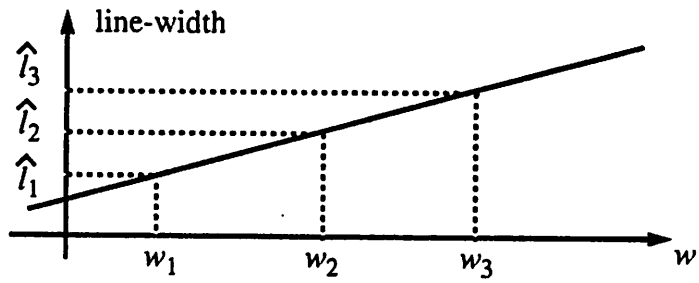
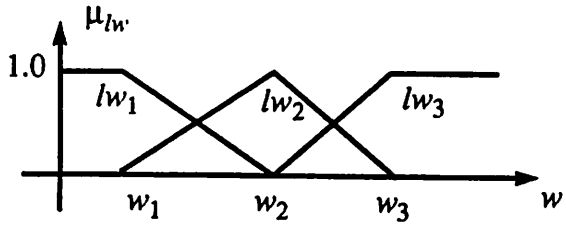


Figure 5.1 Linear Regression Model

where w_1, w_2 and w_3 are the input settings (i.e. = -1, 0, 1 in this case), and $\hat{l}_1$, $\hat{l}_2$ and $\hat{l}_3$ are their corresponding output values predicted by the linear regression model of the form shown in Eq. (2-1) in Chapter 2. The model parameters are extracted from the data using the method of least squares [5.1]. The proposed fuzzy rules are then:

- IF $w = w_i$ THEN $lw = lw_i$. (lw_i are fuzzified linguistic values corresponding to $\hat{l}_i$).

The corresponding membership functions are chosen in the simplest linear form, or the triangular shapes, as shown in Figure 5.2:



lw: Fuzzy linguistic variable

lw_i: Fuzzy linguistic values (i=1,2,3 in this case)

Figure 5.2 Initial Input Membership Functions for Continuous Output Variables

For the categorical variable *mouse bites*, on the other hand, the initial membership functions of p are derived from the logistic regression probability functions (Figure 5.3). The functions were discussed in detail in Chapter 2.

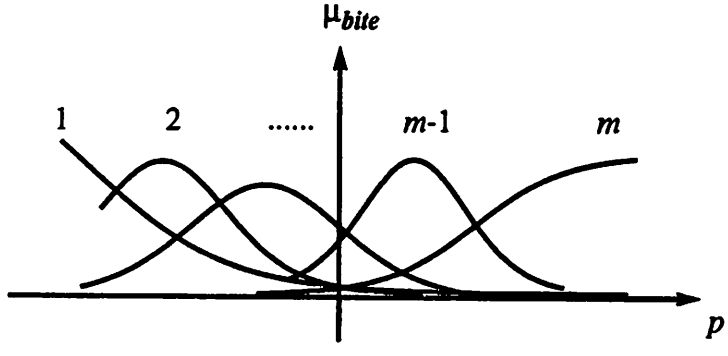


Figure 5.3 Initial Input Membership Functions for Categorical Output Variables

Then the corresponding fuzzy rules would be:

- IF $p = p_0$, THEN *mouse bites* = class m with membership $\mu_i(p_0)$, for i=1,2,...,m.

From Eqs. (2-2), (2-3) and (2-4) in Chapter 2, a maximum value for each probability function can be derived for when $i = 2, ..., m$-1:

$$\max\ p_i(x_i) = p_i(x_i^0) = \tanh\left(\frac{a_i - a_{i-1}}{4}\right), \quad \text{when} \quad b\,x_i^0 = -\frac{a_i + a_{i-1}}{2}, \tag{5-1}$$

where x is the input variable[1]. It is easy to see that these probability functions $p_i(x_i)$ are symmetrical around x_i^0 :

$$p_i(x_i^0 + x) = p_i(x_i^0 - x) \quad , \tag{5-2}$$

Thus, the fuzzy rules for categorical variables can also be written as:

$$\text{IF } X = x_i^0 \ , \ \text{THEN } Y = Y_i \ , \tag{5-3}$$

where X and Y are the (fuzzified) input and output linguistic variables, and Y_i stands for linguistic/categorical value "class i"). These $p_i(x_i)$ will be the initial membership functions for categorical variable such as *mouse bites*. For the boundary cases $i = 1$ and $i = m$ cases, the fuzzy rules become:

$$\text{IF } X = \textit{smallest} \ , \ \text{THEN } Y = Y_1 \ , \tag{5-4}$$

and,

$$\text{IF } X = \textit{largest} \ , \ \text{THEN } Y = Y_m \ . \tag{5-5}$$

Hence, an initial fuzzy inference system can be automatically created from the values of the input recipe settings such as w and p to infer the output parameter values, such as lw

1. The discussion in this section can be easily generalized to multi-input case by substituting x for the input vector x, and $b\,x$ for $b^T \cdot x$.

and *mouse bites*. To make the system self-learning to accommodate newly acquired data, however, we need to define a "cost function" which measures the difference between the experimental data and the model predicted data, as indicated in Chapter 4. Since such a cost function would depend on the membership parameters, an optimization can be carried out to tune the membership functions in the process of self-learning.

For an interval (continuous) variable such as *line-width*, the definition of a cost function is the same as that in Chapter 4 [see Eq. (4-5)], where a self-learning fuzzy inference system was built for the continuous output variables of the semiconductor process. For a categorical output such as *mouse bites*, the corresponding cost function would still be of the same form as for the interval variable:

$$E = \frac{1}{2}\sum_{j=1}^{N}\left(\hat{y}_j - y_j^{\text{meas}}\right)^2 \tag{5-6}$$

However, instead of numerical values, the model predicted output $\hat{y}_j$ and the measured experimental data y_j^{meas} will be two sets of probability values, corresponding to the category values $i = 1, 2, ..., m$, from the fuzzy inference and from the new experimental data, respectively.

To optimize the system and to accommodate newly acquired experimental data, the membership parameters will be changed to minimize the cost function. These adjustable parameters are the centers and widths of triangular membership functions for the interval/continuous variables, and likewise a_i and b_j of the exponential membership functions for the categorical variables.

In Chapter 2, we discussed in detail how to screen out the insignificant input variables based on their corresponding effective test in the statistical regression analysis. After applying this statistical screening [5.2], each output variable is modelled only through the few statistically significant input variables. The corresponding linear regression fitting lines and the logistic regression probability functions can be used to define the initial membership functions of the fuzzy inference system.

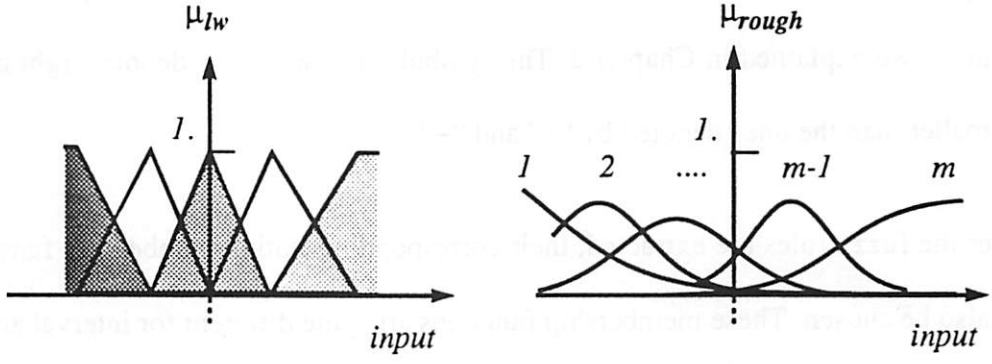


Figure 5.4 Membership Functions

In summary, as shown in Figure 5.4, the membership functions for *line-width* are designed with simple triangular shapes, and they are extracted from the linear regression model. For the ordinal variable *mouse bites*, however, the membership functions have the exponential forms used in the logistic regression model. Such an inference system can incorporate both interval and categorical process responses in the same adaptive set of fuzzy rules. This system will now be described in some detail.

5.2 The Initial Fuzzy System

From the statistical regression analysis, the following rule table is deduced for the data

in Table 5.1

Table 5.2 Statistical Pre-processing

	w	p	O_2
lw	–	0^-	0
rough	+	0^+	0
bite	+	0^+	0

where "+" means output increases when input increases, "–" means output decreases when input increases, and "0" means the output does not show significant dependence on the input, as we explained in Chapter 2. The symbols "0^+" and "0^-" denote slight trends much smaller than the ones denoted by "+" and "–".

After the fuzzy rules are extracted, their corresponding initial membership functions should also be chosen. These membership functions are quite different for interval and for categorical output variables.

5.2.1 Fuzzy inference for interval output variables

For interval variables such as *lw*, the simple triangular membership functions are used. The output values corresponding to the peaks of membership functions are obtained from the linear regression line fitting for singular input variable (Figure 5.5). After the initial rules and membership functions are created, the inference procedure will be the same as those in Chapter 4. Figure 5.6 shows the result of such an inference process when both input variables, power w and pressure p, are considered.

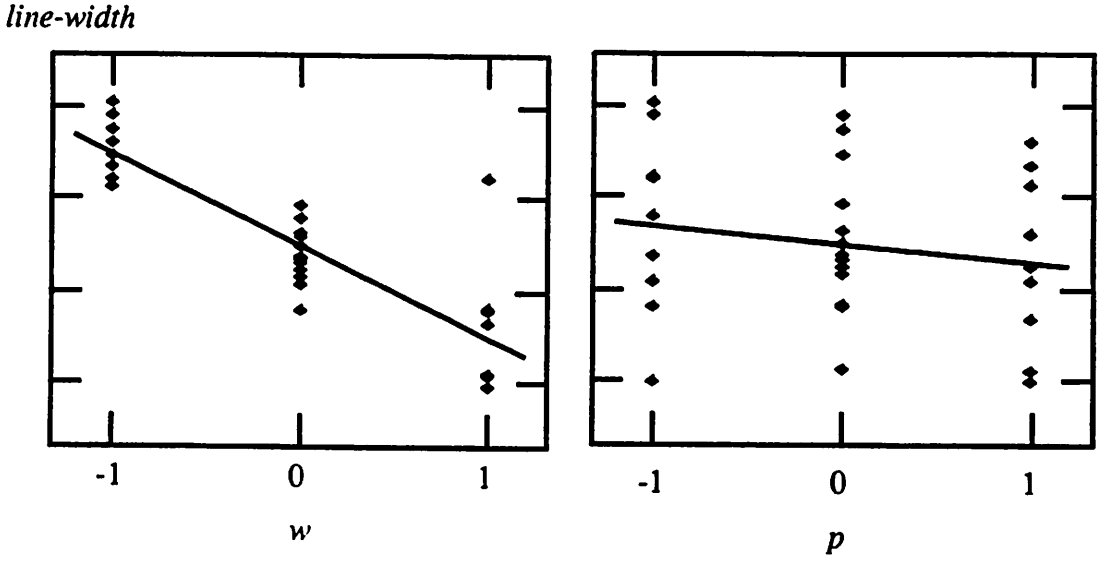


Figure 5.5 Linear Regression Fitting

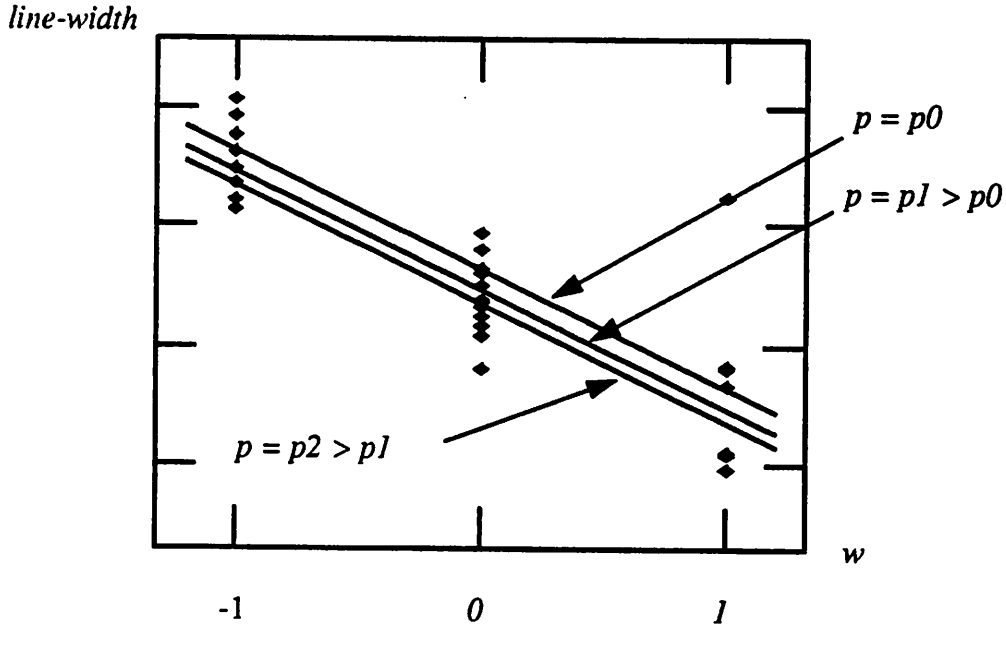


Figure 5.6 *Line-width* vs. *w* for Different *p*

As shown in these figures, *line-width* depends to a large extent on *w* and is only slightly dependent on *p*.

5.2.2 Fuzzy inference for categorical output variables

For the categorical variables *roughness* and *mouse bites*, the probability functions from the logistic regression model are used as the initial membership functions, with their corresponding output values to be the original categorical values. Figure 5.7 shows the probability functions of output *mouse-bite* versus input *w*, after applying the logistic regression analysis.

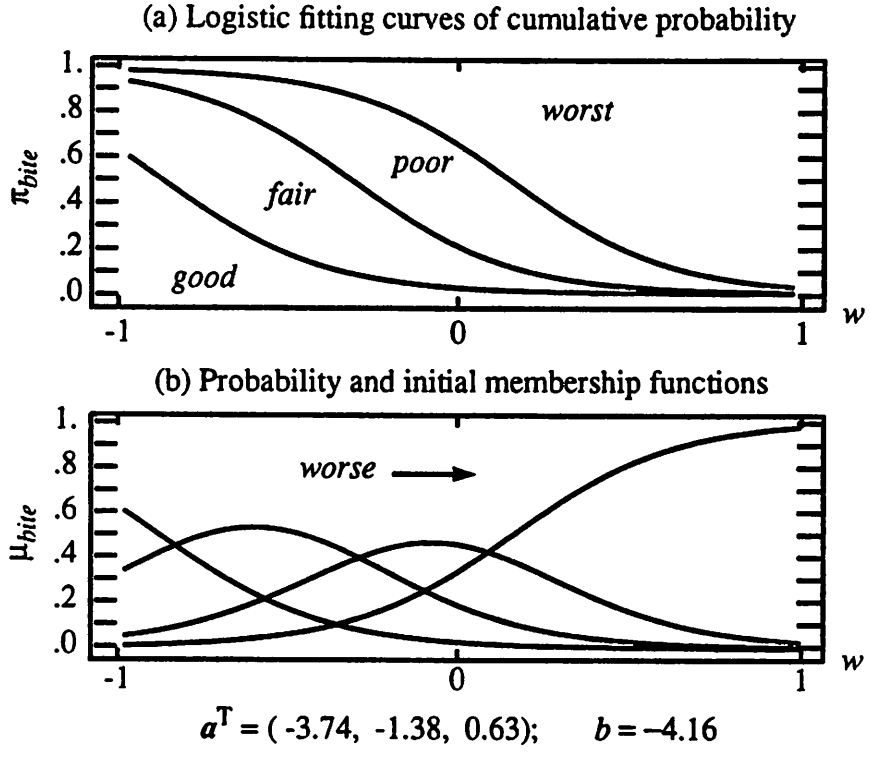


Figure 5.7 *Mouse Bites* vs. *w*

Even though these curves are the same as those in Chapter 2, in the context of the fuzzy inference system they take a new meaning. Namely, instead of probability functions, they now play the role of the initial membership functions. These membership functions

can be used for inference and can be tuned later to accommodate new data. Similarly, Figure 5.8 plots the initial membership functions of *roughness* for all three input variables. Guided by the initial statistical analysis, the effect of the O_2 flow rate is insignificant while the effect of the pressure p is less significant than that of power w.

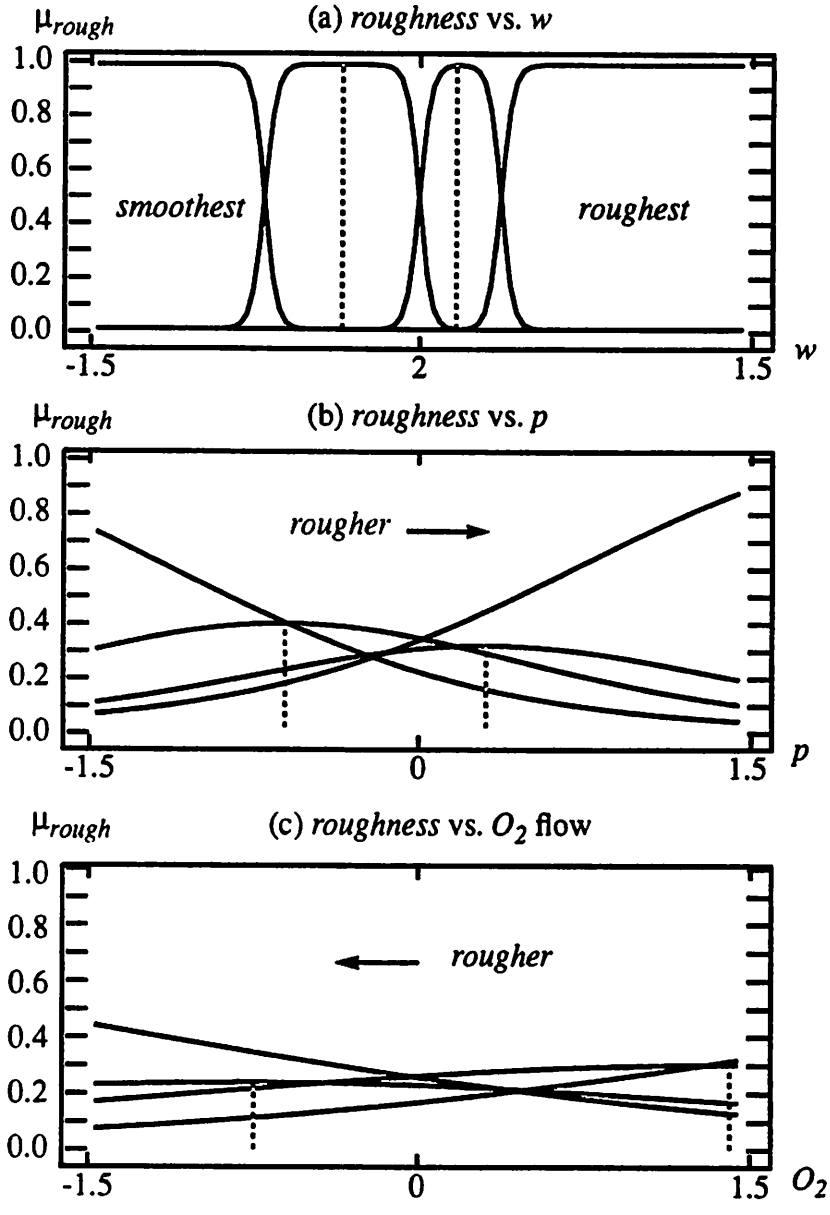


Figure 5.8 *Roughness* vs. *w*, *p* and O_2 Flow Rate

After creating the initial fuzzy inference system, we can apply the fuzzy inference mechanism described in Chapter 4 to derive the output probabilities for *roughness* and *mouse bites* with different input values of power *w* and pressure *p*. The output probability values against input *w* at various values of *p* are plotted in Figure 5.9 and Figure 5.10,

respectively for *roughness* and *mouse bites*.

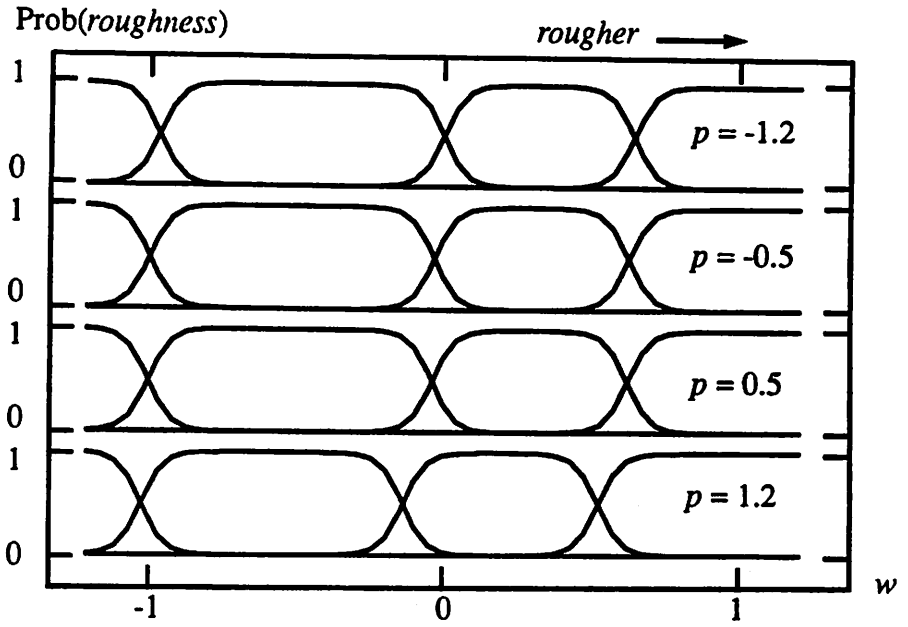


Figure 5.9 *Roughness* vs. *w* for Different Values of *p*

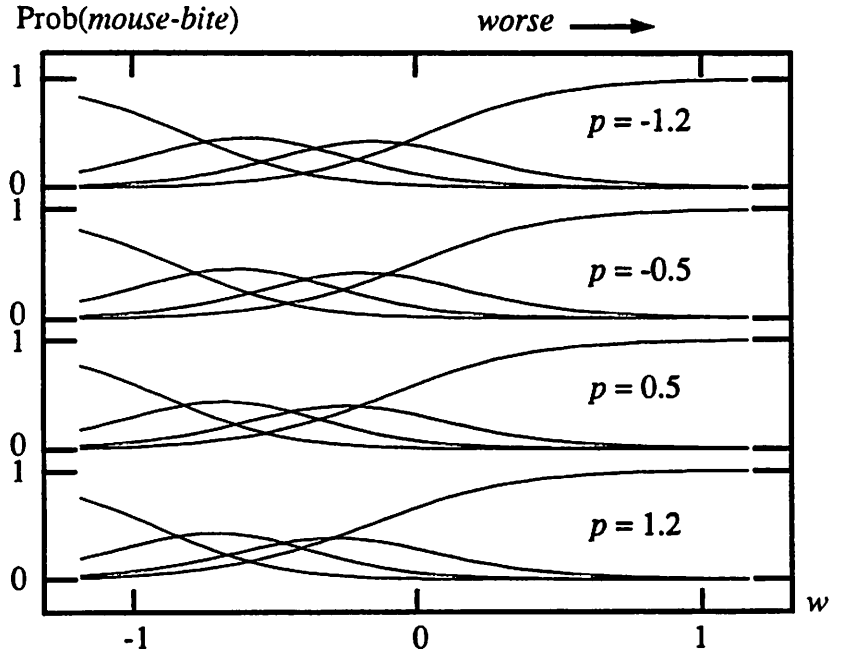


Figure 5.10 *Mouse Bites* vs. *w* for Different Values of *p*

Note that the curves in Figure 5.9 and Figure 5.10 can be interpreted as probability values as well as the membership values of the *output* variable, as they are the results from the fuzzy inference process for different input value combinations for w and p.

5.3 Model Adaptation

To take full advantage of a fuzzy inference system, a self-learning algorithm similar as the one described in Chapter 4 can be used for model adaptation. When applying such a fuzzy inference system designed through categorical data analysis, the shape of the membership functions can be tuned by minimizing the cost function shown in Eq. (5-6).

Figure 5.11 shows the results from such a membership tuning process. We used two different sets (16 data sets for each) of data extracted from the original experiment as our "new" experiment data to tune the system membership parameters. Since the system parameters before tuning are principally based on existing knowledge on a specific process, they should not be changed drastically just to accommodate newly acquired experimental data. Therefore, a modified formula is used in obtaining the results showing in this section.

$$E = \frac{1}{2}\sum_{j=1}^{N}\left(\hat{y}_j - y_j^{\text{new}}\right)^2 + \frac{\text{weight}}{2}\sum_{j=1}^{N}\left(\hat{y}_j - \hat{y}_j^{0}\right)^2 \tag{5-7}$$

where $\hat{y}_j^0$ is the system output prior to the tuning. The modification consists of adding a weighted term to constrain the system not moving too far away from the original state. By minimizing the cost function E, we obtain the updated fuzzy inference system, as shown

in Figure 5.11.

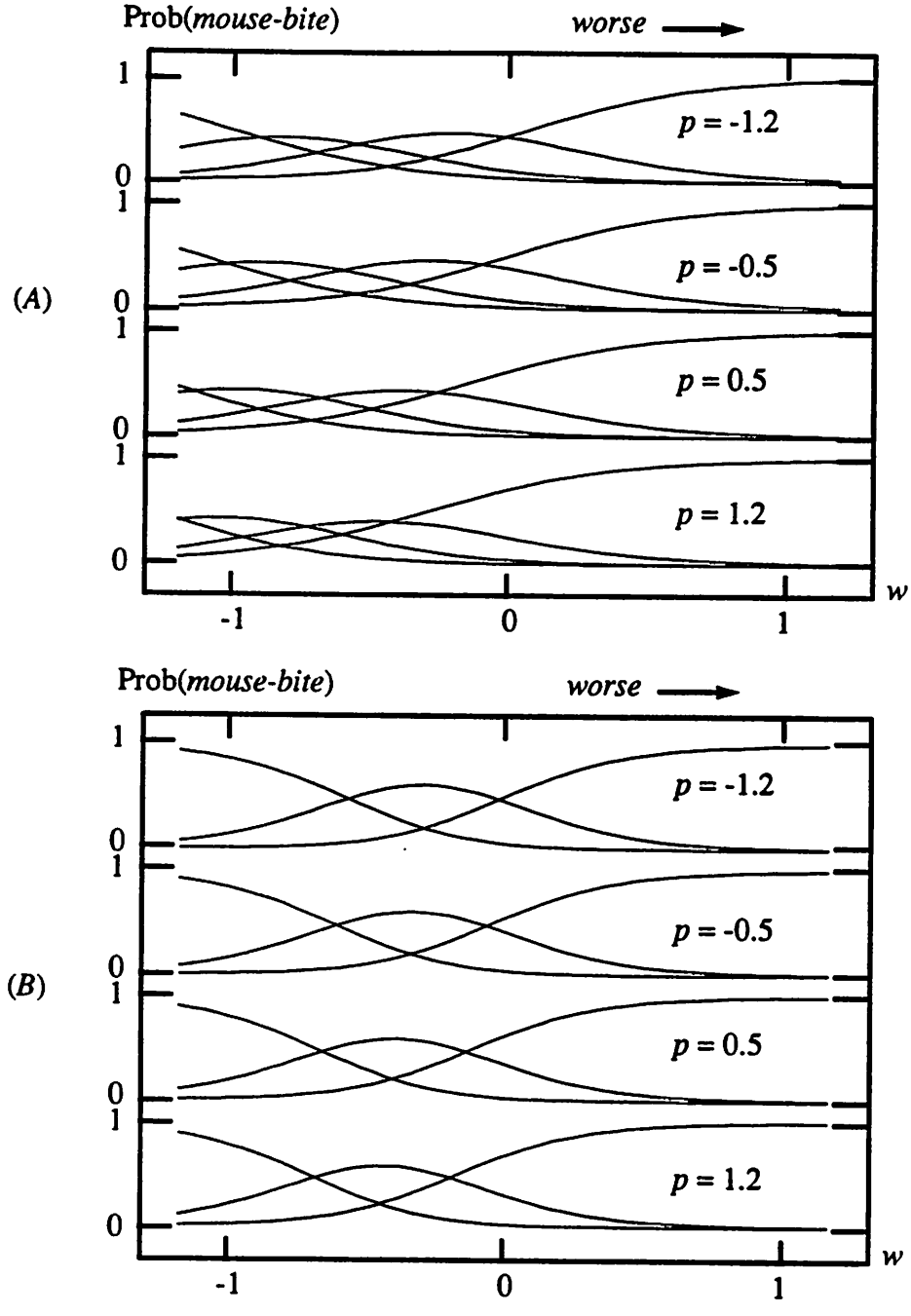


Figure 5.11 *Mouse Bite* vs. w for Different p (Fitted to Data Set A and to Data Set B)

For data set A, we see that the 16 new data points introduced agree quite well with the original system, thus the membership functions show only minor changes. For data set B,

however, the middle two categories ("fair" and "poor") are now merged, indicating a rather drastic change during the learning. The cost function vs. the number of iterations in the optimization process are plotted in Figure 4.11. The same optimization tool is employed as in Chapter 4 [5.3][5.4].

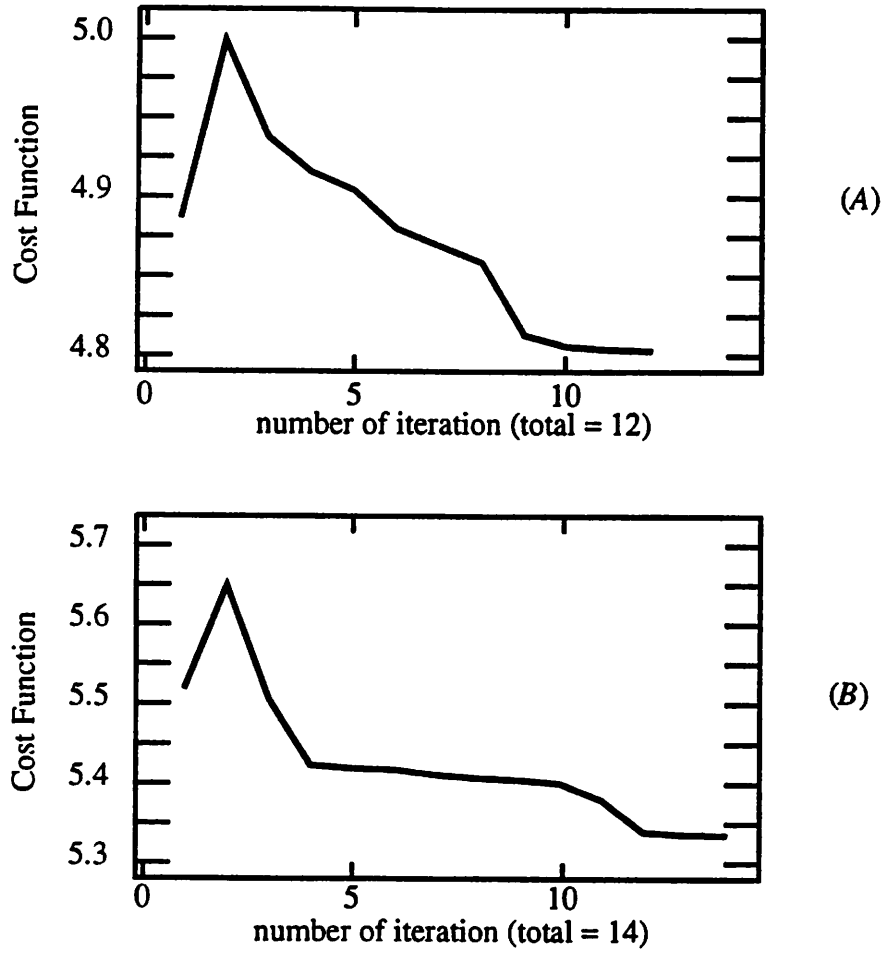


Figure 5.12 Cost Function vs. the Number of Iteration (with data set *A* and data set *B*)

Updating the fuzzy model parameters for interval output variables, such as *line-width*, is the same as shown in Chapter 4. Therefore, the complete system has the additional advantage of integrating interval and categorical output variables.

5.4 A Qualitative Modeling Framework

An object-oriented framework is being built to facilitate both inference system design and system usage. As shown in Figure 5.13, in the system usage period, the fuzzy membership functions will be tuned to accommodate newly acquired experimental data. Such a membership tuning technique is a direct generalization from the methodology in Chapter 4, where only the continuous output variables were considered. A more detailed software implementation of such a framework is discussed in the next chapter.

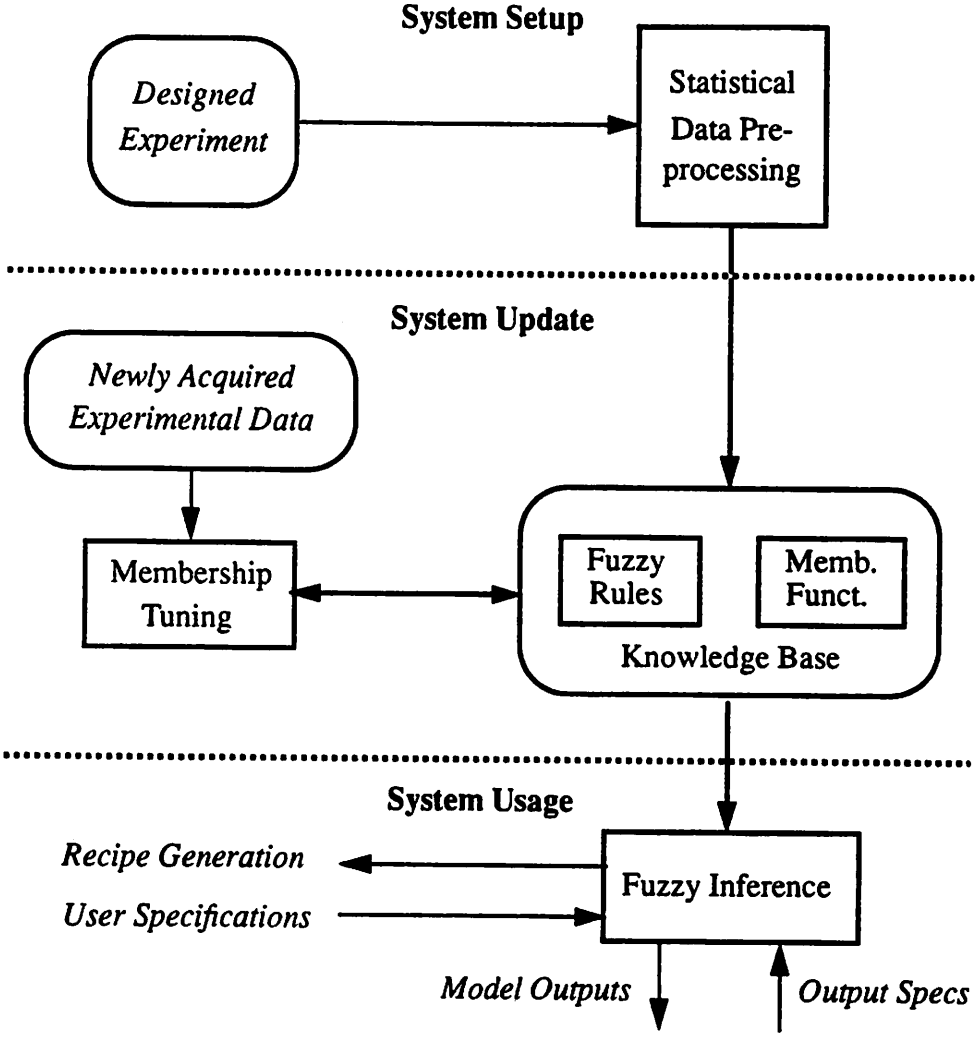


Figure 5.13 Creating, Tuning and Using the Fuzzy Inference System

5.5 Summary

By employing formal statistical data preprocessing, a systematic approach is created to extract fuzzy rules and membership functions to initiate a fuzzy inference system. The statistical regression analysis simplifies the input-output relationships and identifies significant effects in view of experimental noise. This system can accommodate both interval/continuous and categorical/discrete process variables in an integrated format. In addition, it is possible to tune the system to accommodate newly acquired experimental data in a self-learning fashion.

This systematic data pre-processing procedure offers not only better membership functions, but also better fuzzy inference rules extracted from the designed experiments. This is because the system effectively screens out the experimental noise using formal statistical tests. This fuzzy system design approach is well suited to a semiconductor manufacturing environment where process variability and measurement resolution are often comparable to the quantity being monitored.

5.6 References

[5.1] G.E.P. Box, W.G. Hunter and J.S. Hunter, *Statistics for Experimenters - An Introduction to Design, Data Analysis, and Model Building*, John Wiley & Sons, 1978.

[5.2] D.W. Hosmer and S. Lemeshow, *Applied logistic regression*, John-Wiley & Sons, 1989.

[5.3] C.J. Spanos, "Statistical Parameter Extraction for IC Process Characterization," *Ph.D dissertation*, Dept. of Electrical and Computer Engineering, Carnegie-Mellon University (May 1985). (The optimization program package Hanpal is available at BCAM workstation cluster: spanos@radon.eecs.berkeley.edu:prometheus/).

[5.4] M.J.D. Powell, "A Fast Algorithm for Nonlinearly Constrained Optimization Calculation," *Proceedings of 1977 Dundee Conference on Numerical Analysis* (June 1977).

Chapter 6 Software Implementation

In today's semiconductor manufacturing field, software is becoming more and more important both for design and processing. For the target of computer integration of semiconductor manufacturing, we need to develop user-friendly software system to model the semiconductor processing equipment. This is a major objective of the Berkeley Computer Integrated Manufacturing (BCAM) System [6.1][6.2].

The BCAM software system is a framework built to facilitate the experimentation with various CAM applications. It features an interactive user interface that operates in the UNIX X-Window environment and interfaces with various commercial statistical packages and a relational database. The system currently supports real-rime monitoring, real-time SPC, diagnosis, recipe generation, equipment modeling, etc. Most of the existing models, however, cannot capture the qualitative features of a process.

To adequately model a manufacturing sequence, one has to consider the important qualitative aspects of the process. Only a combination of the qualitative and quantitative modeling approaches can produce an effective modeling system which meets modern semiconductor manufacturing requirements.

This chapter will summarize the prototype software implementation of the qualitative modeling approaches presented in previous chapters. They are currently stand-alone software systems in the user account: *raymond@radon.eecs.berkeley.edu:~raymond/*. The final goal, however, is to incorporate these software programs into the BCAM system.

6.1 Self-Learning Fuzzy System — Linear Membership Functions

This software package is written in C language and named "self_learn". The source files are in the directory: ~raymond/programs/learning_demo/. It is a direct software implementation of algorithms developed in Chapter 4. It also interacts with the Hanpal optimizer package, which also serves as the optimizer for the whole BCAM system. Figure 6.1 is the flowchart of the system, where each block represents a major program which carries out the fuzzy inference procedures. More details are given in the source code documentation and in the README file in the directory.

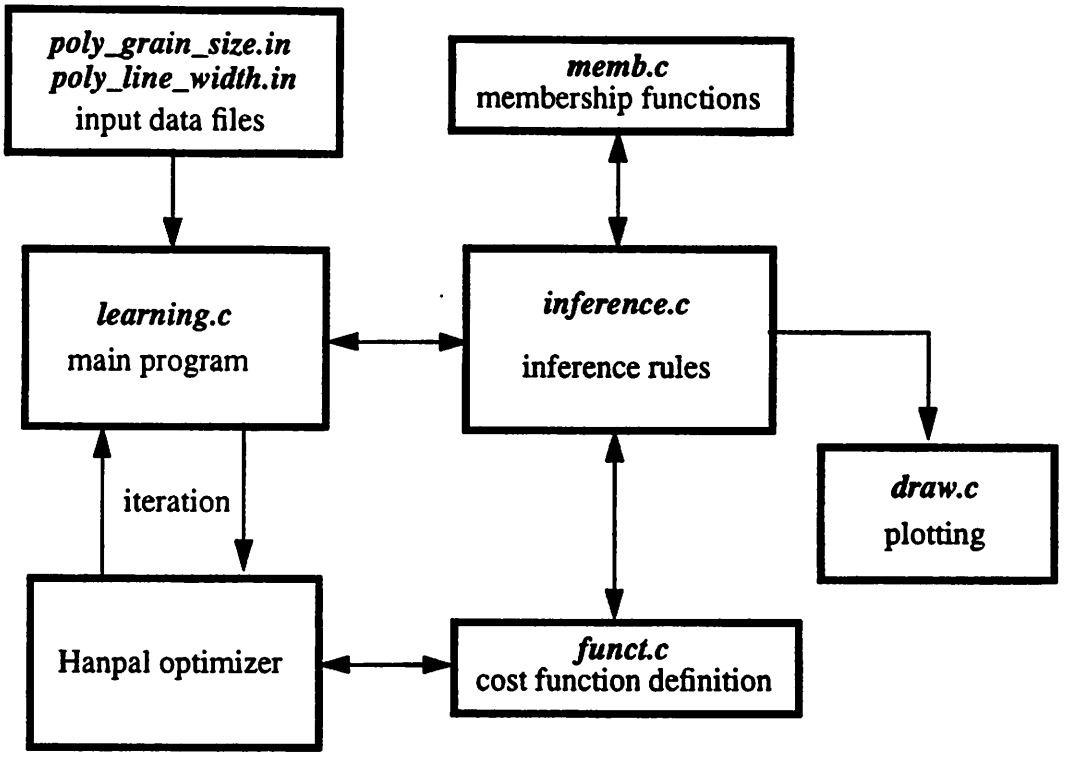


Figure 6.1 Flowchart for Simple Self-Learning Fuzzy System ("self_learn")

There are two types of input files. File *poly_grain_size.in* contains the information for the grain size prediction model discussed in Chapter 4, while *poly_line_width.in* is for the numerical output (e.g. polysilicon line width *lw*) inference model designed with linear

regression analysis (see Chapter 5). All the data for training and tuning the system are stored in the same input file. If applied to a real world problem, however, the tuning data should come from the on-line machine for model adaptation.

6.2 Improved User Interface with C++ and X-Window Programming

These software packages are written in C++ and use the X-toolkit. The source files are in the two directories: ~raymond/programs/inference_demo_c++/ and ~raymond/programs/inference_demo_Xt/. These two packages can only do inference, but they are more flexible and user friendly. The systems accept multiple input factors and multiple output variables. They also allow users to experiment with different shapes of membership functions, such as linear versus exponential. The flowcharts for these two systems are shown in Figure 6.2 and Figure 6.3, respectively.

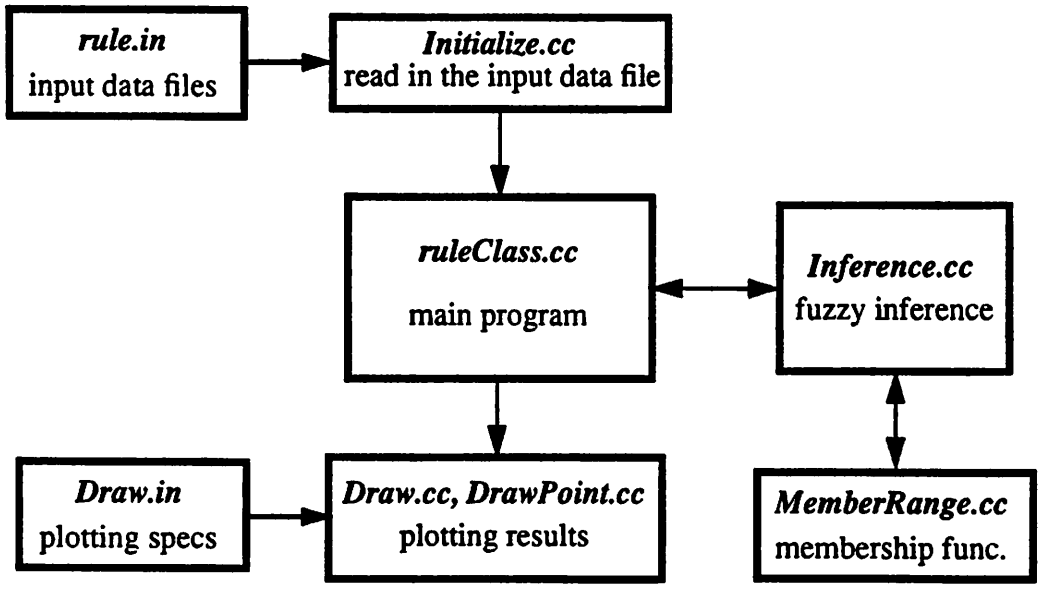


Figure 6.2 Flowchart for *~/programs/inference_demo_c++/*.

There are two *class* objects defined in these two packages, i.e. class *ruleBase*, which includes all the fuzzy rules and membership functions, and class *drawData*, which contains the plotting specs for drawing the inference relationships between the model output variable and input factors. The package "inference_demo_Xt" is similar to "inference_demo_c++", except it has an X-window interface.

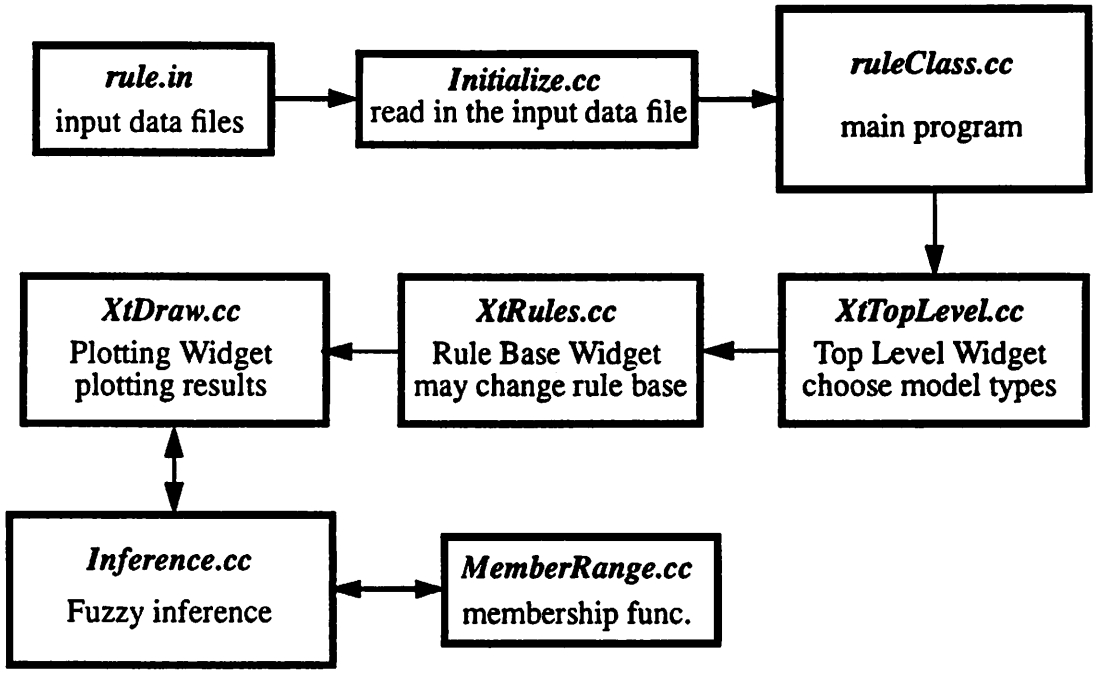


Figure 6.3 Flowchart for *~/programs/inference_demo_Xt/*.

6.3 Fuzzy System Designed with Categorical Data Analysis

One of the main contributions of this project is the combination of fuzzy inference system with the categorical regression analysis, as explained in Chapter 5. There are three steps to realize an adaptive (or self-learning) fuzzy inference system designed with the logit regression model. First, the logistic regression analysis is carried out through the JMP software on Macintosh. The inference part is implemented in the software package

~raymond/programs/logit_demo/ and lastly, the model adaptation programs are in *~raymond/programs/logit_learning_demo/*. This software package is named "logit_learn" and its flowcharts are in Figure 6.4, which uses the *mouse bites* model as an example.

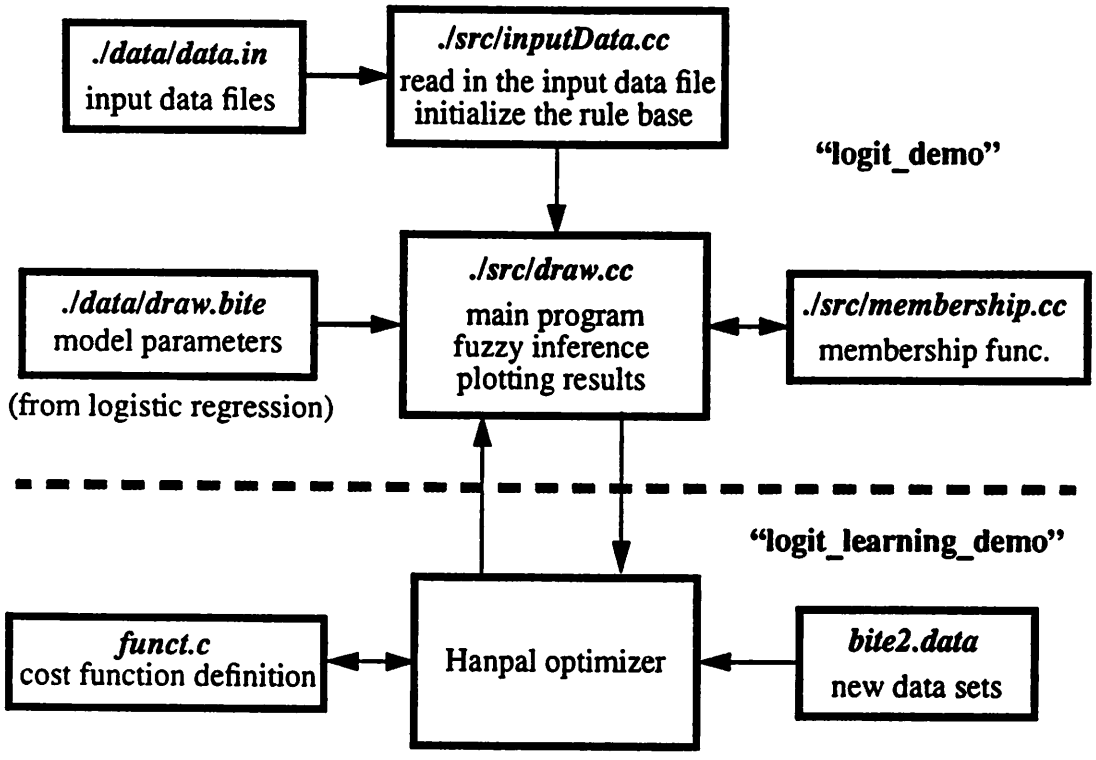


Figure 6.4 Self-Learning Fuzzy System Designed with Logistic Regression

After taking input data obtained from logistic regression analysis, the initial inference system will be created and can be used to predict process output responses for any input setting. When new data sets are added, "logit_learning_demo" will tune the membership parameters through Hanpal optimizer and feed the parameters after tuning back to "logit_demo" for re-run. Since the process output responses for this system are categorical ordinal variables such as *mouse bites* or surface *roughness*, the model outputs are probability distributions for those ordinal variables falling into different categories, as discussed in the text and shown in the figures in Chapter 5.

6.4 Summary

This chapter describes the general program structures for the software packages from which most of the results in the previous chapters are obtained. As Table 6.1 shows, we mainly presented three types of modeling systems, mostly discussed in Chapter 4 and Chapter 5.

Table 6.1 Software Packages for Different Process Output Variables

Package Name	Process Output	System Design	System Inference	Model Adaptation
Self_Learn	*Grain Size*	human designated	*learning_demo/*	*learning_demo/*
Self_Learn	*Line-width*	*statistical pre-processing*	*learning_demo/*	*learning_demo/*
Logit_Learn	*Mouse Bites*	*statistical pre-processing*	*logit_demo/*	*logit_learning_demo/*

More detailed programming informations are given in the Appendix in the dissertation, the *README* files in each software package, and comments included in individual input data files and program files. Further improvements of the software implementations are discussed in the next chapter.

6.5 References

[6.1] C.J. Spanos, "The Berkeley Computer-Aided Manufacturing System," *SRC/DARPA CIM-IC Workshop*, North Carolina State University (August 1991).

[6.2] B.J. Bombay, "The BCAM Control and Monitoring Environment," *M.S.E.E. Thesis*, Memorandum No. UCB/ERL M92/113, Electronics Research Laboratory, University of California at Berkeley (September 1992).

Chapter 7 Conclusions And Future Work

We have presented a systematic methodology as well as an implementation of the algorithm that takes advantage of the qualitative aspects of the semiconductor manufacturing process. This work offers a novel approach to characterize and apply the important, yet qualitative knowledge of a process. Within our modeling framework, categorical process output responses can be modeled along with quantitative responses. The conventional application of statistical process control can be generalized for those qualitative process variables. The main modeling approach, however, is a fuzzy inference system which captures the human knowledge on the qualitative aspects of a process. Such a system can be designed with the logistical regression analysis with categorical data, used with the fuzzy logic base inference rules, and updated with a self-learning scheme by tuning the membership functions. Some actual semiconductor processes are chosen as test vehicles for such a modeling framework, including the Low Pressure Chemical Vapor Deposition (LPCVD) process and the Plasma Dry Etching process. The results show a promising future for such a qualitative modeling methodology.

Our final goal is to establish, within the Berkeley Computer-Aided Manufacturing (BCAM) system [7.1][7.2], a qualitative modeling module which extracts automatically qualitative knowledge from designed experiments and updates the knowledge database for newly acquired data. Such a module can be used for equipment and process simulation, recipe generation, diagnosis, and control. Some discussions on the possible future work based on this Ph.D. project are presented in this chapter.

7.1 Possible Future Projects

7.1.1 Qualitative Knowledge Base and Data Base

Additional qualitative models can be developed for more equipment and processes in the future. When the fuzzy inference system becomes a BCAM module, it should share the same database, collected from real processes, with other modules running within the BCAM system [7.2]. There could be also a database for the fuzzy rules and other qualitative knowledge, available to the BCAM users. This will help the user to build new qualitative models for some specific processes or update the existing models.

7.1.2 Fuzzy Recipe Generation

The qualitative inference module can lead to a fuzzy recipe generation module, where a manufacturing process recipe for input-settings will be generated from user specifications of outputs [7.3]. Those outputs can be qualitative as well as quantitative in nature. This recipe generation procedure is a reverse inference problem, since the manufacturing process outputs (user specs) become the inputs of the fuzzy system, therefore both exponential shape (initialized from logistic regression) and triangular shape (initialized from linear regression) of membership functions are to be considered together at the same time. In order to create appropriate membership functions, however, it is expected to require a great amount of process data to build a reliable recipe model.

7.1.3 Optimization Algorithm

Our results show that the fuzzy logic based inference system can well represent qualitative knowledge and adaptively adjust the system parameters to fit new data. In a real-world problem, however, the input variables are likely inter-dependent, and the member-

ship tuning process might become too complicated to solve with a generic optimizer such as the Hanpal used in this work [7.4][7.5]. Therefore, more investigation may be carried out to find the best optimization tools for our fuzzy inference system. Some possible approaches might include the Neural Network [7.6][7.7] and/or Genetic Algorithm [7.8][7.9] methods.

7.1.4 Software Integration

As shown in Table 6.1, the current version of software implementation are stand-alone. In order to integrate the software packages discussed in Chapter 6 under a single systematic module framework, several improvements should be made [7.2][7.10]:

- to run the categorical analysis as well as the linear regression analysis in the UNIX environment, then the results could automatically be used to create the initial fuzzy inference system.
- to upgrade the Hanpal optimizer with object-oriented C++ programming, so the model adaptation can be integrated with the rest of the system.
- To integrate the fuzzy inference system with the rest of the BCAM system, including sharing the same data base, library function, and X-window interface.

And finally, to apply such a qualitative modeling methodology in the industry, a great amount of work on testing and reliability investigations ought to be carried out, which requires industry cooperation with the supply of additional process data and expertise.

7.2 References

[7.1] C.J. Spanos, "The Berkeley Computer-Aided Manufacturing System," *SRC/ DARPA CIM-IC Workshop*, North Carolina State University (August 1991).

[7.2] B.J. Bombay, "The BCAM Control and Monitoring Environment," *M.S.E.E. Thesis*, Memorandum No. UCB/ERL **M92**/113, Electronics Research Laboratory, University of California at Berkeley (September 1992).

[7.3] K.K. Lin, "Modeling and Characterization of Semiconductor Manufacturing Equipment: An Application to LPCVD Reactors," *Ph.D. Dissertation*, Memorandum No. UCB/ERL **M90**/44, Electronics Research Laboratory, University of California at Berkeley (May 1990).

[7.4] C.J. Spanos, "Statistical Parameter Extraction for IC Process Characterization," *Ph.D dissertation*, Dept. of Electrical and Computer Engineering, Carnegie-Mellon University (May 1985). (The optimization program package Hanpal is available at BCAM workstation cluster: spanos@radon.eecs.berkeley.edu:prometheus/).

[7.5] M.J.D. Powell, "A Fast Algorithm for Nonlinearly Constrained Optimization Calculation," *Proceedings of 1977 Dundee Conference on Numerical Analysis* (June 1977).

[7.6] B. Kosko, *Neural Networks and Fuzzy Systems*, Prentice Hall, 1992.

[7.7] R. Ichikawa, K. Nishimura, M. Kunigi and K. Shimada, "Auto-Tuning Method of Fuzzy Membership Functions Using Neural Network Learning Algorithms,"

Proceedings of the *2nd International Conference on Fuzzy Logic and Neural Networks*, pp. 345-48, Japan, 1992.

[7.8] S. Rodriguez, A. Paricio and J. Velasco, "Learning with Fuzzy Logic: A Way to Combine Genetic Algorithms and Fuzzy Logic," Proceedings of the *NASA International Workshop on Neural Networks and Fuzzy Logic*, pp. 165-74, 1992.

[7.9] T. Espy, E. Vombrack and J. Aldridge, "Application of Genetic Algorithms to Tuning Fuzzy Control Systems," Proceedings of the *NASA International Workshop on Neural Networks and Fuzzy Logic*, pp. 237-48, 1992.

[7.10] C.J. Spanos and D.C. Mudie, *private communications*, 1994.

Appendix User's Guide for Software Packages

There are five software packages developed during this project. The computer user account: *raymond@radon.eecs.berkeley.edu:~raymond/programs/*, contains a sub-directory of ~raymond/programs/ that includes a set of programs. There are *README* files in each sub-directory which describes the purpose of the package and the steps to execute the programs. There are also a *READ.file_list* file for each package which lists the files in that package. The name of each package is similar to that of the sub-directory that contains it. Some of the general notations for all packages are listed as follows:

- The first part of all input files contains the values read by the program. The second part contains the corresponding variable names in the programs for these input values.

- In each group of data values, the first line indicates the name for the variable (a character string), the second line is the number of values for that variable to be entered (the length of a data array). What follows is the value of each variable.

- Variable names starting or ending with *min, max, step* in series are usually specs for plotting, i.e. number of *steps* within the range [*min_value, max_value*].

- Within the programs, the definition and the explanation of a variable is given in the header files, where the variable is first declared.

- Most output data files are plotting files for the UNIX command *xgraph*. Most of the figures in this dissertation were imported from these *xgraph* plots.

A.1 Simple Fuzzy Inference and Adaptation — *learning_demo* ("self_learn")

This package is a simple self-learning fuzzy inference system. Here "simple" means that the membership functions are triangular. It was used to produce the results discussed in Chapter 4 (grain size model) and in Chapter 5 (line width model), using the two input files: *poly_grain_size.in* and *poly_line_width.in*, respectively. For the grain size model, the system reads in the experimental data sets directly from the input files, and input values of these experimental data sets are transformed directly to the membership parameters. For the line width model, however, the input files contain results from the linear regression analysis of the original experimental data, which are used to build the triangular membership functions. After building the membership functions, the system treats these two cases identically for model inference and model adaptation.

The adaptation or self-learning part of the program is done with the *Hanpal* optimizer. The Makefile first makes the object file *funct.o* and the *Hanpal* library function *libfunct.a* by combining the cost function definition *funct.c* with the rest of the *Hanpal* optimizer package in user directory: *~spanos/prometheus/*. The rest of the programs are mainly used for fuzzy inference, and for plotting the results with the UNIX *xgraph* utility. The two programs, *random.c* and *systems.c*, are testing programs created in the model development process for verifying the inference algorithm, and they are not needed during normal execution. The file *memb.c* evaluates the membership function values and can be treated as part of the main inference program: *inference.c*, which carries out the fuzzy inference operation. The file *draw_point.c* is for drawing points on *xgraph* figures with different styles, and should be treated as part of the main plotting program *draw.c*, which takes the

plotting specs (input/output names, plotting ranges for inputs, plotting points for each curve, etc.) from the input files *grain_size.in* or *linear_regression.in*. And finally, the program *learning.c* serves as the main program for reading in the input data, organizing different subroutine programs and plotting outputs. The relevant output files are:

- *optim.out* — log file from the Hanpal optimization process.
- *hanpal.in1*, *hanpal.in2* — input parameter files for Hanpal optimizer.
- *hanpal.data*, *learning.output* — output parameters from Hanpal.
- *hanpal.plot* — cost function versus the number of iteration, e.g. Figure 4.11.
- *membership.plot* — membership function plots before and after optimization, e.g. Figure 4.12.
- *systems.plot* — system inference results plots, e.g. Figure 4.10.

A.2 Fuzzy Inference Systems Designed with Logistic Regression — *logit_demo* ("logit_learn")

The package contains a fuzzy inference system with membership functions designed from logistic regression analysis for categorical data, such as *mouse bites* (see Chapter 5). The programs are written in C++ object-oriented programming and are structured in several sub-directories: *bin/* for all the object file and executable files (binary files), *include/* for head files, *src/* for all the C++ source code, *data/* for the input data files and *output/* for all the output files that can be plotted using *xgraph*.

In addition to the usual *README*, *READ.file_list* and *Makefile* files in this package,

there is an alias file called *Runfile*. It is designed for making the running of this software package easier. The notes in the *README* file explain the alias names used for running this software with *Runfile*. For instance, "*rund dr22*" is an alias name for UNIX shell command "*make -f ~raymond/programs/logit_demo/Runfile dr22*", and then *dr22* is an object defined in that *Runfile* file.

There are two types of input files: *data.in* is the original experimental data table and *draw.<out>* is the drawing specs and the membership parameters resulted from logistic (as well as linear) regression analysis, where *<out>* is the name of process output variable, such as *mouse bites*. Another input file in that sub-directory *data/* is called *draw.<out>_learning*, which contains membership functions resulting from tuning with the *Hanpal* optimizer (see Section A.3).

There are also two kinds of output plotting files: *<data_table_#>.<output>_<input>* contains the experimental data sets read in from the input file *data.in*, and *<data_table_#>-<output>_<input>* are the results from the model inference, where *<data_table_#>* stands for the label of the experimental data table (only data table #2 is explained is this dissertation), *<output>* is the process output name (e.g. *bite* stands for *mouse bites*), and *<input>* is the process input name (e.g. *w* stand for the power *w*). Those output plotting files are shown as Figure 5.5 to Figure 5.11 in Chapter 5. Note that this system also includes an inference model for the numerical process output *line-width*. This demonstrates that we can incorporate both numerical and categorical output models in the same fuzzy inference system.

A.3 Model Adaptation for the Fuzzy Inference Systems Designed with Logistic Regression — *logit_learning_demo* ("logit_learn")

This software package is complementary to the package "*logit_demo*", i.e. it tunes the membership functions for the fuzzy inference system "*logit_demo*" through minimizing a cost function with the Hanpal optimizer. The cost function is defined in *funct.c*, and follows Eq. (5-7) in Section 5.3. The input file *<out>.data* (e.g. *bite.data*) contains the initial membership parameters and the new experimental data sets with which the system is to be optimized. The output membership parameters after tuning are written in the file called *draw.<out>_learning*, which will be copied in the input sub-directory for the software package "*logit_demo*" for re-running the inference model with the tuned membership functions. Another output file, *hanpal.plot*, is the plotting file for the cost function values versus iteration numbers (e.g. Figure 4.11).

There are two equivalent methods to compare the model results with the experimental data shown as follows.

Suppose there are three model probabilities p, q, r, corresponding to the three categories P, Q, R of a process output categorical variable, and $p+q+r = 1$. And the newly acquired data sets are results are from $k+m+n$ experimental runs with k, m, n times falling in categories P, Q, R, respectively. Then we may have two methods in constructing the cost functions. We can, for example, make the experimental probability data for each category based on only one experimental run, i.e. either 1or 0, depending whether or not the output falls into that specific category. Therefore, the first choice of cost function would be in the following format:

- $E_1 = (p - 1)^2 + (q - 0)^2 + (r - 0)^2$ — assume the first run falls in category P,

  $+ (p - 0)^2 + (q - 1)^2 + (r - 0)^2$ — and another run falls in Q,

  $+ \ldots\ldots$ — and so on .....

  $= k (p - 1)^2 + (m + n) p^2 + m (q - 1)^2 + (n + k) q^2 + n (r - 1)^2 + (k + m) r^2$

  $= (k+m+n) (p^2 + q^2 + r^2) - 2 (k p + m q + n r) + (k + m + n)$

On the other hand, since the model prediction of the probability values are numbers between [0,1] for each category, we may also group together the experimental data having the same input setting in order to have an input probability value between 0 and 1. Therefore, we may have an alternative definition of the cost function with the experimental probability values are $k/(k+m+n)$, $m/(k+m+n)$, $n/(k+m+n)$, for categories P, Q, R, respectively:

- $E_2 = [p - k/(k+m+n)]^2 + [q - m/(k+m+n)]^2 + [r - n/(k+m+n)]^2$

  $= (p^2 + q^2 + r^2) - 2 (k p + m q + n r)/(k+m+n) + (k^2 + m^2 + n^2)/(k+m+n)^2$

Therefore, the difference between E_1 and E_2:

$$E_1/(k+m+n) - E_2 = 1 - (k^2 + m^2 + n^2)/(k+m+n)^2$$

$$= 2 (k m + m n + n k)/(k+m+n)^2$$

is a constant, which means that these two methods are equivalent in optimization with the minimization of the cost function. For the convenience of programing, we choose the method in E_1 as our cost function.

To make the Hanpal optimizer effective and convergent, we have to introduce several

constraints which reflect the properties of the membership functions initially derived from logistic modeling. These constraints are explained in the forms of inequality functions defined in the cost function definition file *funct.c* and are listed below:

- Three membership parameters α_1, α_2, α_3 are originally the three intercepts of logit model (see Chapter 3 and Chapter 5), hence they should be sequential in value:

  $$\alpha_{min} < \alpha_1 < \alpha_2 < \alpha_3 < \alpha_{max}$$

  where the optimization range $[\alpha_{min}, \alpha_{max}]$ should be given by the program user.

- The maximum value point for the two middle membership functions represent the two linguistic values (*mouse bites* model) *fair* and *poor*. They should still remain within the original input range after tuning [see equations (5-1) to (5-5) in Chapter 5].

- The membership parameters β should not change sign during the optimization, since that would imply a reversion in the direction of dependency between the input and output variables.

A.4 Simple Fuzzy Inference Systems with *C++* Objected-Oriented Programming — *inference_demo_c++*

This software package is an upgraded version of the inference part of package "*learning_demo*", with C++ objected-oriented programming and flexibility for multiple inputs and outputs. There are two kinds of C++ *class* objects. The first one, *ruleBase*, is used for storing fuzzy rules and membership functions, and the other one, *drawData*, is used for storing the plotting specs. Similarly, input file *rule.in* inputs fuzzy rules and membership functions to the system, and another input file *Draw.in* inputs the specifications for

plotting the inference results. Output plotting files are in the form of *system.plot_<in>*, where *<in>* stands for the main process input variable (e.g. deposition temperature T_d for the grain size model). The membership functions can be chosen by the users to be linear or exponential, specified in *Draw.in*.

A.5 Simple Fuzzy Inference Systems with X-window Interface — *inference_demo_Xt*

This software package is an upgrade from the package "*inference_demo_c++*", with the addition of UNIX X-window toolkit programming (Athena Widgets). The default input date file *rule.in* inputs the original fuzzy rules and membership functions, which are editable through the X-window interface widgets when programs are in running. The plotting specs can also be typed in through those widgets dynamically. The user can use a number of different types of membership functions for better simulation results.

There are three levels of interface widgets, implemented with programs *XtTopLevel.cc*, *XtRules.cc* and *XtDraw.cc*, respectively. The top level widget specifies the input/output names and the types of membership functions to be used (e.g. linear or exponential). The second level widget, or the rule table widget, specifies the rule base along with the membership function parameters. Finally, the last level widget is for plotting the model inference results.