LITHOGRAPHY PARAMETER EXTRACTION AND OPTIMIZATION USING NEURAL NETWORKS

By

Luca Cazzanti

A thesis submitted in partial fulfillment of the

requirements for the degree of

Master of Science

(Electrical Engineering)

at the

UNIVERSITY OF WISCONSIN – MADISON

1998

To my parents, Mario and Maria

Acknowledgements

I would like to thank my advisor, Prof. Franco Cerrina, for teaching me how to conduct scientific research. I am deeply indebted to him, as he contributed greatly to my formative years in engineering.

I am also grateful to Prof. Yu Hen Hu, for the helpful discussions on neural networks and for always being available for meetings.

My parents, Mario and Maria, have supported me throughout my college years with their understanding and encouragement. To them goes my deepest gratitude.

Finally, I thank Susie, for her love and patience.

This work was supported in part by grants from the Semiconductor Research Corporation under Grant Number 96-LP-452, the National Science Foundation under Grant Number DMR-95-31009, and DARPA/ONR under Grant Number N00014-96-1-0588.

List of Figures

List of Tables

Contents

Chapter 1

Introduction

The fabrication of semiconductor devices involves the repetition and combination of complex processes. Because of the complexity and costs, a simple "trial-and-error" approach to process design is often infeasible. However, the power of modern computers allows us to construct software models that simulate the response of the processes within acceptable time [1, 2].

The importance of computer simulations lies in their ability to predict the outcome of manufacturing processes and to provide insight into the underlying physical phenomena that are being modeled. This ability becomes even more essential in advanced research, because often the process cannot be realized physically due to limitations in the capabilities of current manufacturing equipment.

In the context of process modeling it is often required to perform *parameter extraction*. Given a process of the form:

$$\mathbf{y}(x_1, \ldots, x_n; p_1, \ldots, p_m) = F_{p_1 \cdots p_m}(x_1, \ldots, x_n) \tag{1.1}$$

we distinguish between *input variables* $(x_1, \ldots, x_n)$ and *control parameters* $(p_1, \ldots, p_m)$. Starting from a set of specifications on the process

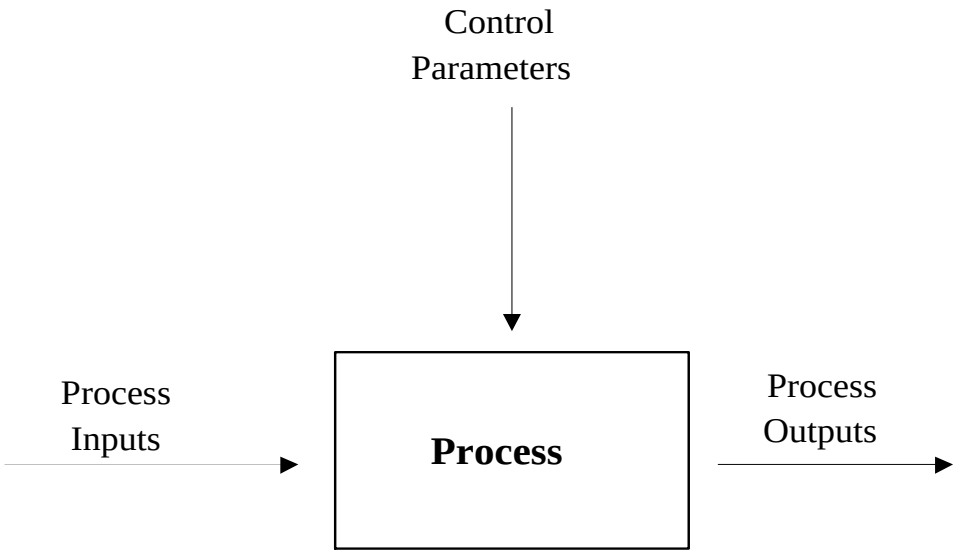

Figure 1: *Dependency of the process outputs on the process inputs and control parameters.*

response, we wish to know how to control the process parameters in order to guarantee that the outputs will satisfy the given specifications, i.e. $\mathbf{y}(x_1, \ldots, x_n; p_1, \ldots, p_m) \in \mathbf{y_o} \pm \triangle\mathbf{y}$.

The extracted parameters in general will define a subset of the possible range of the process parameters. It is useful and often necessary to perform simple statistical analyses on the elements of the subset and obtain bounds on the *process latitude*, which is a measure of the level of freedom available in choosing the values of the parameters [3].

This research explores the application of *artificial neural networks (NN)* to the parameter extraction problem. As an example, we select the case of semiconductor manufacturing processes based on X-ray lithography. In the recent years, a large body of literature has described

the ability of NNs to model very complex systems and to act as universal function approximators [4, 5, 6]. In semiconductor manufacturing, NNs have been utilized successfully for modeling and controlling processes based on plasma etching and reactive ion etching [7, 8, 9, 10, 11, 12, 13]. A previous study in X-ray lithography also indicated that the NN approach to parameter extraction is more efficient than other numerical techniques currently used [14]. This research intends to expand the previous work on NNs in X-ray lithography by exploring empirical rules for efficient parameter extraction based on artificial neural networks.

Chapter 2

Statement of Problem and Background

2.1 The Problem of Parameter Extraction

While the approach is general, we will focus the discussion on X-ray Lithography. In particular, we consider the process of transferring a desired pattern to a silicon wafer. In a typical setup for X-ray lithography, an X-ray source illuminates a pattern-carrying mask. The pattern on the mask is defined in a material which either absorbs or transmits X-rays. The resultant modulated radiation field, modified by diffraction, is then recorded on the silicon wafer [1] (Fig. 2).

Typical parameters that influence the characteristics of the recorded pattern include the *thickness* and *material* of the absorber on the mask, and the distance (*gap*) between the mask and wafer. By controlling these and other relevant parameters, it is possible to achieve a desired recorded pattern.

Controlling the process parameters, however, is a non-trivial task.

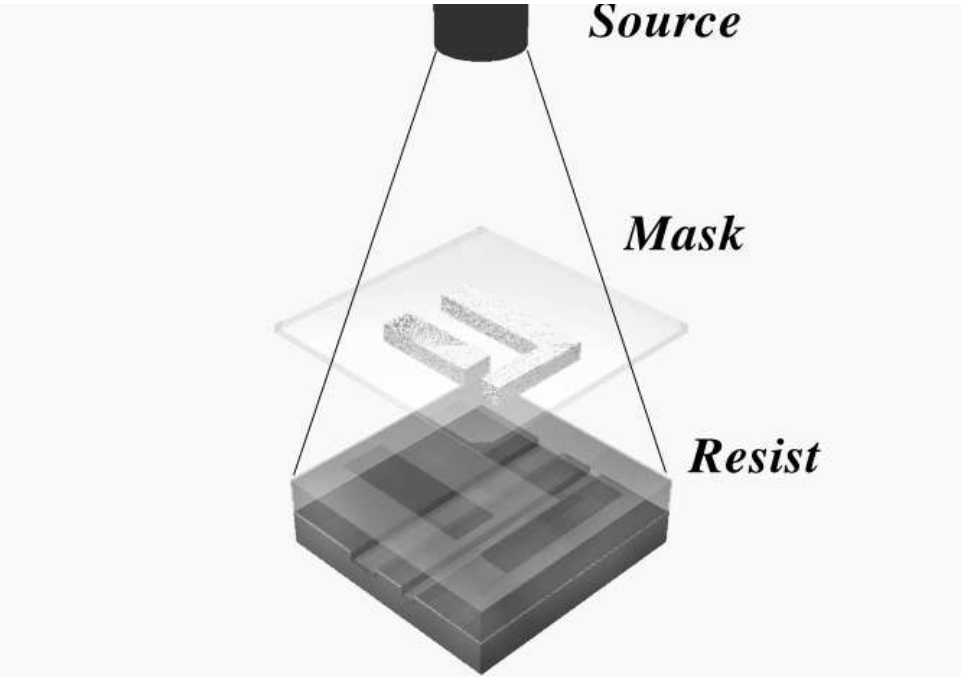

Figure 2: *Typical X-ray lithography setup.*

Diffraction effects, coupled with the interaction of X-rays with the materials in use, render the process highly non-linear, making an explicit formulation of the relationship between the parameters and the recorded pattern essentially impossible [1].

Since an explicit formulation of the process is not known a priori, it is convenient to view the process as a *mapping from an input space to an output space.* To this end, we postulate the existence of a function

$$\Phi(x_1, x_2, \ldots, x_n; p_1, p_2, \ldots, p_m) \Longrightarrow (y_1, y_2, \ldots, y_q) \qquad (2.1)$$

which maps a vector $\mathbf{x} \in \mathbf{R^n}$ of process inputs to a vector $\mathbf{y} \in \mathbf{R^q}$ of process outputs, according to some control parameters (vector $\mathbf{p} \in \mathbf{R^m}$).

In the framework of the implicit mapping formulation, we can thus state the **parameter extraction problem**:

> *Given a subset $\tilde{Y} \subseteq Y$ of process outputs which satisfy a set of manufacturing requirements, identify the subset $\tilde{P} \subseteq P$ of control parameters such that $\phi(x_1, \ldots, x_n; \tilde{P}) \Longrightarrow \tilde{Y}$ for the "largest" $\tilde{X} \subseteq X$.*

Figure 3 shows a pictorial view of the parameter extraction problem for the case of three dimensional control parameters and outputs spaces. In general the problem is formulated in R^m and R^q. Artificial neural networks will be the tool used to solve the parameter extraction problem.

2.2 Previous Work

Many software packages, developed by both academic and commercial institutions, perform simulations relevant to semiconductor manufacturing. PROLITH is an optical lithography modeling suite which simulates formation of an image by a projection optical system (pattern transfer), and exposure and development of the transferred image [15]. SAMPLE3D also models pattern transferring [14], while SUPREM

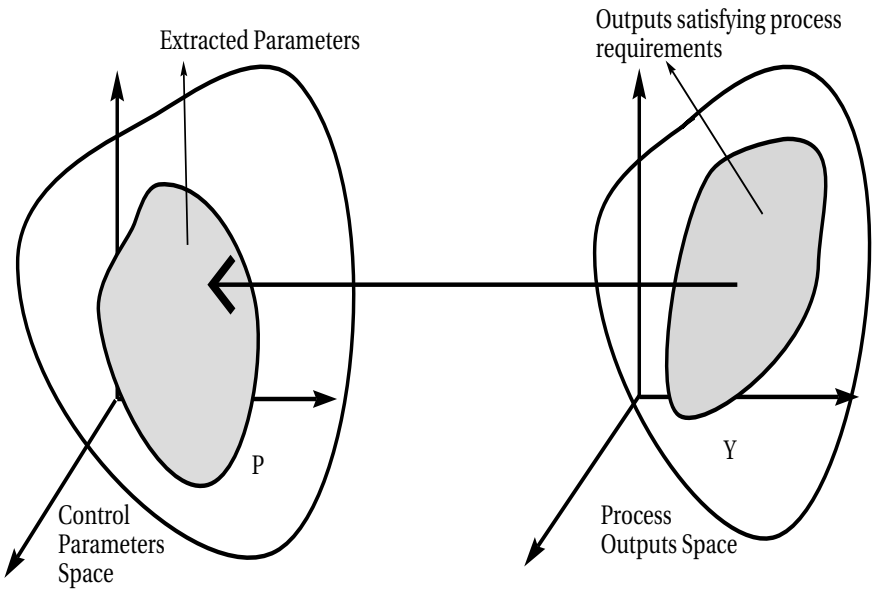

Figure 3: *Pictorial view of the parameter extraction problem.*

[16]simulates ion implantation and diffusion, and PISCES device modeling. The ATLAS/ATHENA suites are integrated systems which simulate the entire fabrication process from pattern transfer, to wafer processing, to the final integrated device [17, 18].

The CXrL ToolSet, developed at the Center for X-ray Lithography of the University of Wisconsin–Madison, is a suite of modeling tools which simulate the pattern transfer process in the case of X-ray lithography. It combines modeling and design tools for the X-ray source, pattern transfer, and wafer processing [1, 19]. Simulations executed on the ToolSet provided the data used to develop the neural network models of this research.

The software packages described above do not implement parameter extraction techniques with artificial neural networks. In fact, often parameter extraction is performed by calculating process outputs corresponding to many different process conditions on a "trial-and-error" basis. However, the semiconductor manufacturing community has already reported successful applications of neural networks to process modeling and control, particularly in the fields of plasma etching and reactive ion etching. These results encourage us to pursue the neural network approach, using X-ray lithography as a test case.

Baker, Himmel and May [13] developed a neural network model of a reactive ion etching process. The model was used to predict the future values of the control parameters based on their past values and their corresponding outputs. The prediction capability was integrated into an alarm system which warns the human operator of any "out of range" conditions.

Rietman and Lory [9] and Rietman [10] report successful neural models of plasma etching processes. They illustrate the prediction capabilities and the small error of neural network-based models. In [11], Han and May described a method by which NNs are used to model plasma-enhanced chemical vapor deposition (PECVD) processes. These models are then used in combination with genetic algorithms to synthesize a process "recipe" for parameter extraction and control. In [12], Nami,

Misman, Erbil, and May illustrate a general technique for constructing a "hybrid" neural model for PECVD, by integrating an abstract input-output mapping with prior knowledge of the process under investigation.

In the case of X-ray lithography, Capodieci [14] compared the performance of neural networks models with other techniques, such as genetic algorithms, simulated annealing, and random search. The neural network approach allowed Capodieci to correctly model the process with less samples of the input-output mapping than the other techniques, while achieving a better error performance.

The reasons for the success reported by the literature rest in the properties of artificial neural networks. May [2] and Himmel and May [20] explain the advantages of using NNs over other numerical techniques in semiconductor manufacturing. NNs are able to model very complex systems involving many process variables, whereas statistical models are not able to perform adequately when the number of variables increases beyond a few. Furthermore, a neural network is not limited by a priori assumptions on the input-output relationship (linear, quadratic, cubic, and so on), but rather it is said to "learn" these relationships by *training* itself on examples of input-output mappings (non-parametric modeling) [21].

An important property of NNs is their ability to act as *universal function approximators*, provided that enough data points are collected.

Several authors have independently published mathematical proofs of this statement [4, 5, 6]. The issue of collecting enough data points is an active area of research. While there exist techniques for determining the size of the data set in the case of statistical process modeling, formal procedures in the case of NNs have not been developed [2]. This research will investigate the sampling issue in later chapters.

Chapter 3

Proposed Solution

3.1 The Process as an Input-Output Mapping

As described in the previous chapter, artificial neural networks perform well on problems involving many variables which may be related to each other by complicated non-linear relationships. In particular, neural networks "learn" underlying relationships between the process variables from a set of input-output mappings which act as "examples" of the process. Therefore, in order to exploit the learning capabilities of neural networks, we must first formulate the parameter extraction problem as an input-output mapping.

There is a distinction between the inputs and outputs of the NN and the inputs and outputs of the process. In our formulation, we can "train" a neural network to learn the *dependency of the process outputs on the control parameters.* The NN inputs are the control parameters of the process, and the outputs are the process outputs. The process inputs are kept constant in the NN formulation so they do not appear

explicitly in the mapping, but rather are taken into account implicitly. Effectively, the NN approximates the unknown function

$$G(p_1, \ldots, p_m) \mid_{x_1, x_2, \ldots, x_n} \Longrightarrow (y_1, \ldots, y_q) \tag{3.1}$$

It is important to obtain an approximation to (3.1), because it could offer a computationally inexpensive method of obtaining process outputs. After learning from a set of example data points, the NN is able to compute the process outputs in a fraction of the time normally required by the current simulation tools of the CXrL ToolSet [14]. Therefore, it becomes more efficient to carry out an exhaustive search of the control parameters space in order to find a range of values which will guarantee that the outputs will satisfy certain manufacturing conditions. Using this methodology, we can extract process parameters as follows:

1. *Use the NN approximation to the direct process to obtain more process outputs as a function of the control parameters.*

2. *Search the outputs thus obtained for those responses that satisfy certain conditions imposed by the user.*

3. *Examine the control parameters corresponding to the outputs found above.*

4. *Infer general characteristics of the extracted control parameters.*

After the problem is formulated as an input-output mapping, we need to design a neural network that will approximate it. The next

Process Variables		*NN Model*
Inputs	$\longmapsto$	Fixed
Control Parameters	$\longmapsto$	Inputs
Outputs	$\longmapsto$	Outputs

Table 1: *Role of the process variables in the NN model.*

section briefly describes the general procedure for designing a neural network. For more detailed descriptions of neural networks, the reader is referred to [22, 23].

3.2 Neural Network Design

In a typical neural computing project, we must collect the data, "teach" the neural network the task that we want to perform, and finally test the designed network. A series of computer simulations carried out on the CXrL ToolSet provides the data. The data collected must contain enough information about the process for the NN to approximate the input-output mapping. Ideally, many data points are to be provided. However, the simulations necessary to obtain the data are time-consuming, so some effort should be made to minimize the number of simulations necessary to design the NN. The issue of collecting an appropriate set of data points (i.e. the problem of sampling) will be discussed in chapter 6.

The collected data is split into two disjoint sets, the *training set*

and a smaller *testing set*. During the training phase, the NN "learns" the mapping from inputs to outputs from the training set. In order to asses how well the network has learned the mapping, a measure of error is adopted. If $(y_1, y_2, \ldots, y_q)$ is the vector representing the outputs in the original data, and $(\tilde{y_1}, \tilde{y_2}, \ldots, \tilde{y_q})$ is the vector of approximations to the outputs provided by the NN, we define the *sum squared error SSE* as:

$$SSE = \sum_{j=1}^{q} (y_j - \tilde{y_j})^2 \tag{3.2}$$

Training stops after the SSE has reached a required minimum value. In order to understand how the training process tries to minimize the SSE, we must briefly discuss the structure of artificial neural networks.

Haykin [22] defines a neural network as follows:

> *A neural network is a massively parallel distributed processor that has a natural propensity for storing experiential knowledge and making it available for use. It resembles the brain in two respects:*
>
> 1. *Knowledge is acquired by the network through a learning process.*
> 2. *Inter-neuron connection strengths known as synaptic weights are used to store the knowledge.*

The parallel structure of neural networks is illustrated in figure 4. The

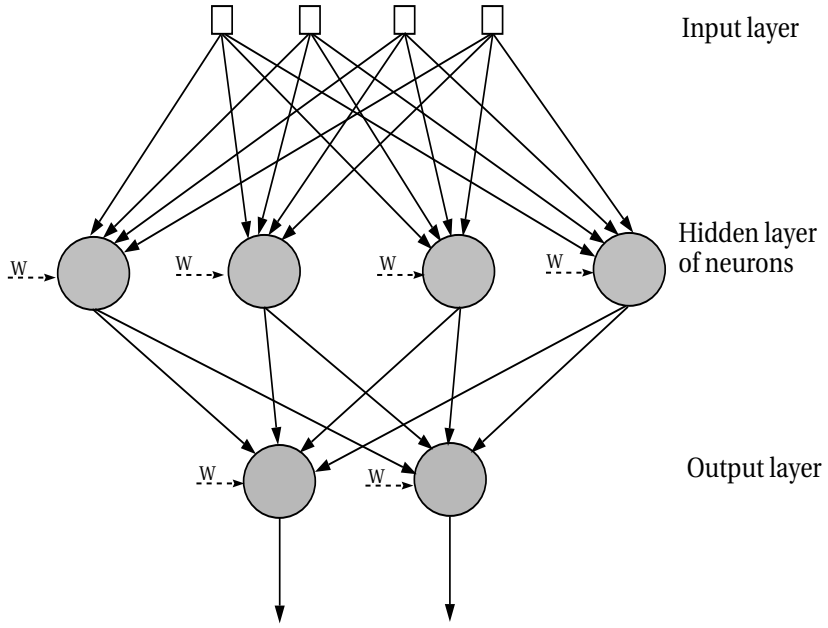

Figure 4: *Structure of a typical neural network.*

inputs in the *input layer* are fed to a set of processing units (*neurons*) in the *hidden layer*. The outputs of the hidden neurons are then processed again by neurons in the *output layer*. The neurons process incoming information by applying a non-linear *activation function* to linear combinations of the inputs, through the synaptic weights W. *Learning consists of updating the synaptic weights so to minimize the SSE*. Usually, the activation function of the output layer is linear, resulting in a neural network which is a linear combination of non-linear activation functions acting upon the inputs. Activation functions commonly used are:

- logarithmic sigmoid: $f(x) = \frac{1}{1-e^{-x}}$

- hyperbolic tangent: $f(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$
- radial basis: $f(x) = e^{\frac{-(c-x)^2}{\sigma^2}}$

When the SSE falls below a target threshold, the neural network has learned the mapping on the training set. However, the network must be tested on the testing set to verify that the NN mapping is general to the entire process under investigation, and not limited to the training set. The inputs from the testing set are applied to the trained network, and the outputs are compared with the corresponding outputs in the testing set. The SSE may be used to evaluate the performance of the network. It is also useful to adopt other measures for the error, namely the *point-by-point error* and the *point-by-point percent error*:

$$e_j = |y_j - \tilde{y_j}| \tag{3.3}$$

$$e'_j = |\frac{y_j - \tilde{y_j}}{y_j}| \times 100 \tag{3.4}$$

The learning and testing steps are repeated until good generalization is achieved (Fig. 5).

The next chapters will present applications of the concepts and techniques introduced here. Neural networks will be used to extract process parameters in the case of a pattern produced by X-ray lithography. We will specify the inputs and the outputs to the neural network, the network architectures used, and the results achieved.

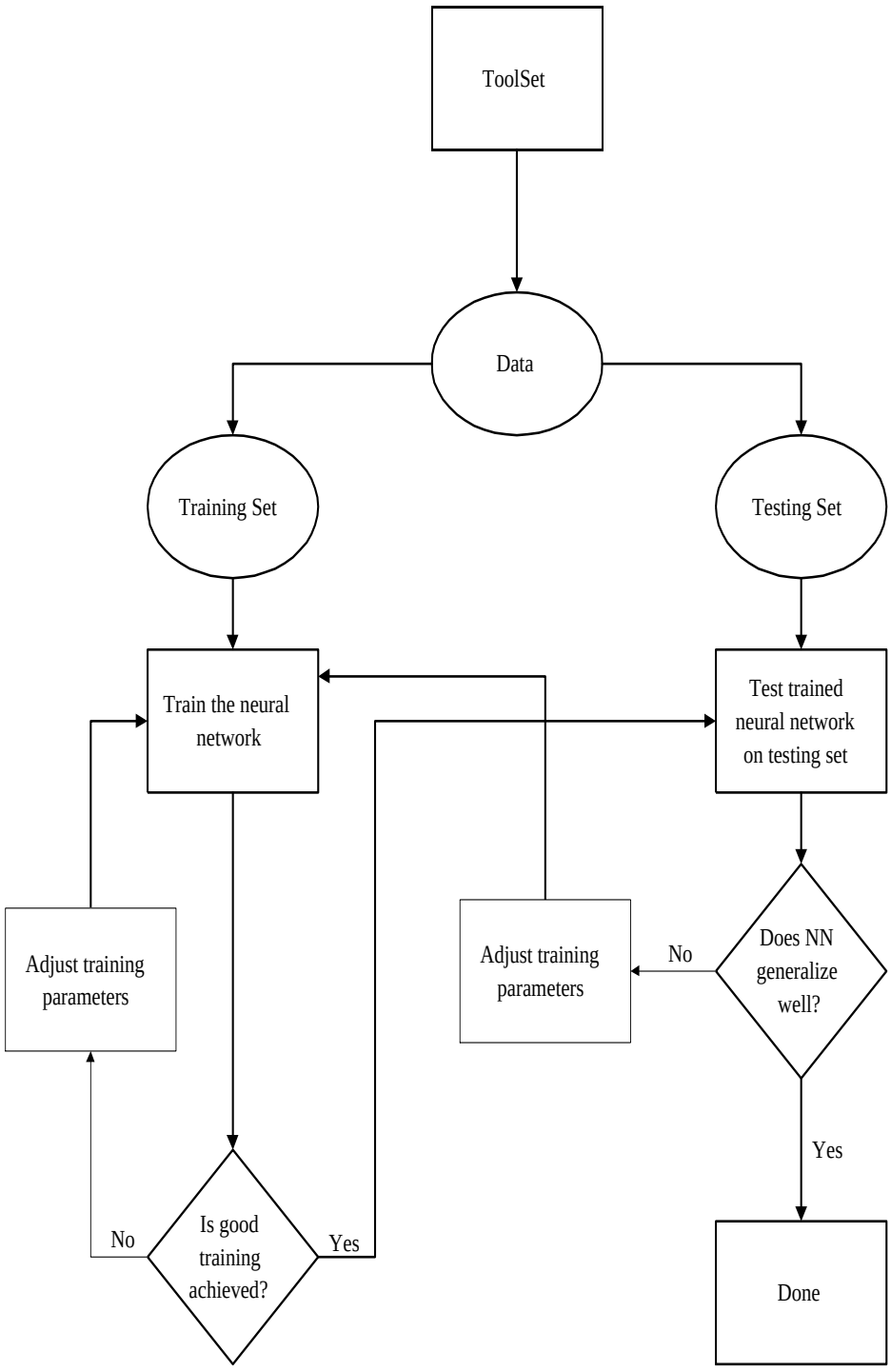

Figure 5: *The training procedure for a neural network.*

Chapter 4

Implementation of a Case Study

4.1 Description of Case Study

A case study was analyzed in order to evaluate the performance of neural networks on the parameter extraction problem. The data were collected from simulations carried out on the CXrL ToolSet. We first define the parameters involved in the simulations, and then describe the NN implementation.

Figure 6 depicts the set-up modeled in the case study. X-rays illuminate a mask consisting of a 2 μm thick silicon nitride (SiN) membrane and a binary pattern of absorbing material (tungsten). The binary pattern is a sequence of lines, 0.13 μm wide, separated by spaces, also 0.13 μm wide. This lines/spaces pattern is repeated 8 times, making the total width of the pattern 2.08 μm. The resulting intensity profile propagates in Helium through the gap. The blur parameter is a lumped quantity used to account for noise and vibration during the exposure

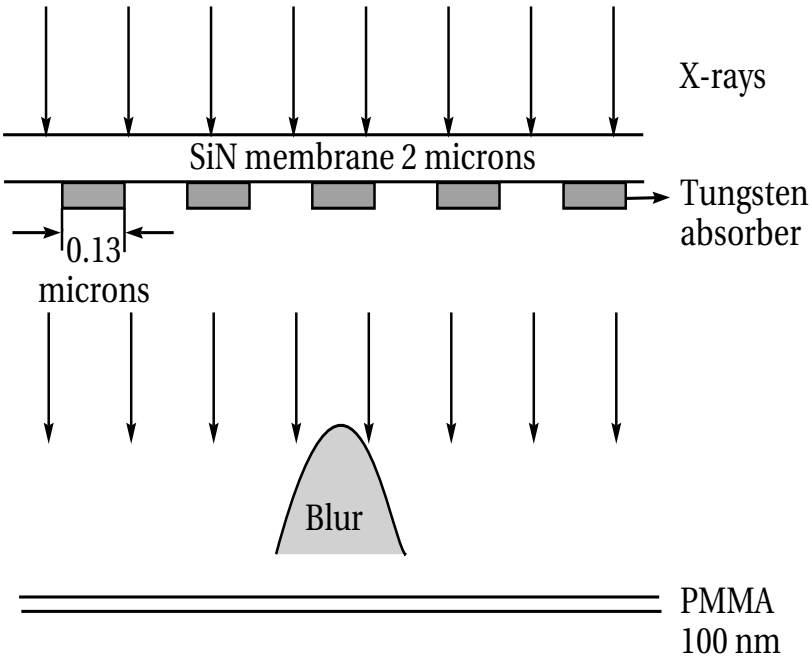

Figure 6: *The case study under investigation.*

[1]. It is expressed mathematically as the variance of a Gaussian distribution, and in the case study it assumed a fixed value of 26 nm. Finally, the modulated intensity hits the surface of the silicon wafer, where a layer of photoresist material is encoded with the resultant *aerial image* pattern (Fig. 7). The photoresist used was PMMA, with a thickness of 100 nm.

The parameters described above constitute the input parameters of the process, and from the point of view of the neural network implementation, they are kept constant. The control parameters, however,

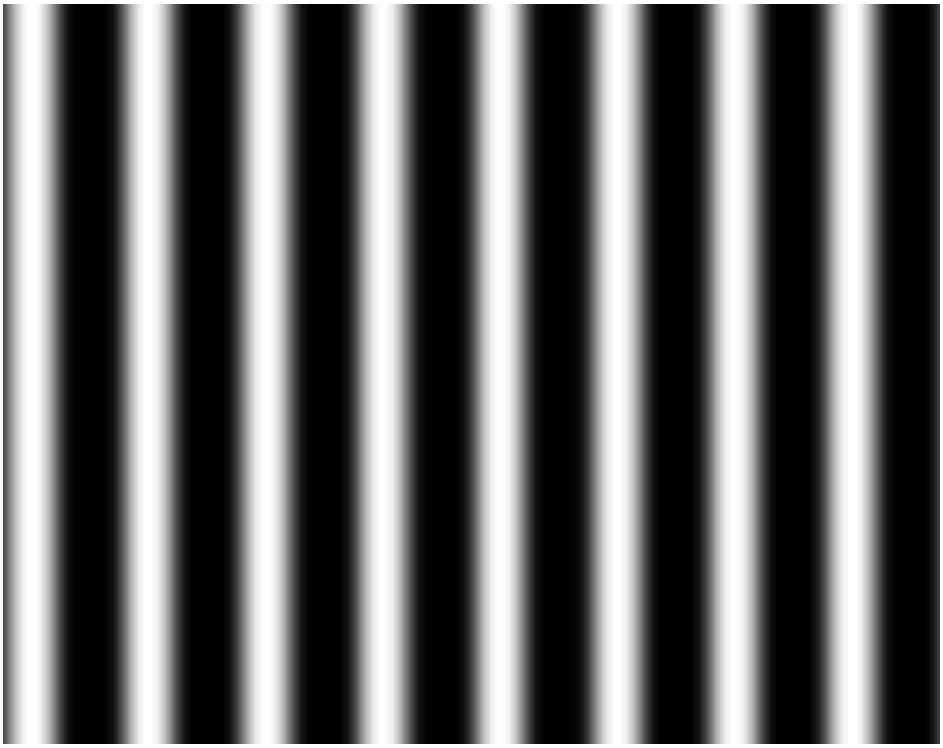

Figure 7: *A sample aerial image encoded in 100 nm of PMMA.*

Input (Fixed) Parameters	
Membrane	SiN, 2 μm thick
Absorber	Tungsten
Pattern	Lines/Spaces, 0.13 μm
Blur	26 nm
Photoresist	PMMA, 100 nm
Control Parameters	
Absorber Thickness	200 to 400 nm
Gap	10 to 30 μm
Bias	-18 to 26 nm
Output Parameters	
Linewidth (LW)	
Fidelity	

Table 2: *Summary table of the parameters in the case study*

are changed during the simulation run and the resultant process outputs are observed. The control parameters used in the case study were the *thickness of the absorber*, the *gap between the mask and the wafer planes*, and the *bias*, which is a parameter used to control slight variations in the width of the absorber. Table 2 summarizes the values of the parameters used in the case study.

In order to evaluate the effect of the control parameters on the outputs, two figures of merit (FOM) were used: the *linewidth (LW)* and the *Fidelity*. The LW is computed by thresholding the intensity distribution of a cross-section of the aerial image at the photoresist plane (Fig. 8). The fidelity may be intuitively described as an average of the pixel-by-pixel difference between the aerial image and a target image

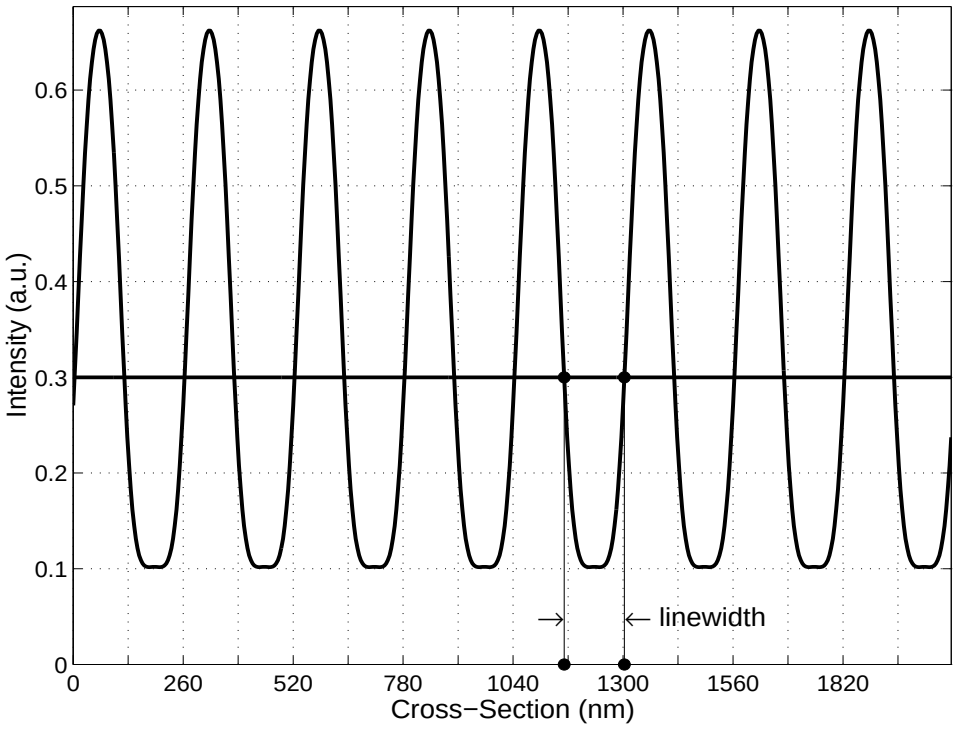

Figure 8: *Linewidth measurement from a cross-section of the intensity profile.*

which is considered "ideal." If we denote the intensity distribution of the target and aerial images by $T(x, y)$ and $R(x, y)$ respectively, the mathematical expression for the fidelity is [24]:

$$F = 1 - \frac{\int\int (T(x,y) - R(x,y))^2 \, dx \, dy}{\int\int T(x,y)^2 \, dx \, dy} \tag{4.1}$$

where 1 is the best possible value and -∞ is the worst.

The rationale for using the LW and Fidelity as the FOMs of choice is that they provide information about *both* the physical dimension and the quality of the pattern in the resist. The combination of these two FOMs allows us to decide which outputs are desirable and which are

to be rejected. Correspondingly, the control parameters that generate desirable outputs are deemed acceptable, while the others are rejected.

4.2 Neural Network Implementation

After obtaining the data from the ToolSet, we may design a neural network that approximates the mapping:

$$(THICKNESS, GAP, BIAS) \Longrightarrow (LW, FIDELITY) \qquad (4.2)$$

Effectively, we construct a neural mapping that models the *dependency of the process outputs on the control parameters*, while keeping the inputs constant. The neural mapping allows us to expedite the process of obtaining many process outputs. A simulation run on the ToolSet takes from a few minutes to several hours to compute, so obtaining thousands of process output values is cumbersome. The neural network allows computation of many process outputs in a few seconds.

Since the neural network offers a computationally efficient method of computing a large number of process outputs, it is possible to exhaustively search numerous responses for the values which satisfy certain requirements. In particular, we can search for the LW/fidelity pair which is closest to a nominal value (*critical dimension, or CD*), while achieving the best possible quality. The values of the control parameters corresponding to the best outputs will be the *extracted values*.

Often, we are not interested only in an approximation to the "optimum" values of the parameters, but also in the *range* of values that will guarantee that the outputs will fall in an acceptable range. For example, we can search for the values of the LW which are within $\pm 10\%$ of the target CD, and for the values of the fidelity which are closest to 1 (the best possible value). The control parameters corresponding to such outputs will define a subset $\tilde{P}$ of the entire control parameter space P.

Analyzing the distribution of the parameters within $\tilde{P}$ helps us understand how much freedom we have in choosing the parameter values while still achieving acceptable outputs. Again, the neural network allows us speed up the step of acquiring enough data points to perform analyses of the physical process.

Chapter 5

Case Study Results

5.1 Neural Network Performance

A neural network was constructed to approximate the mapping $(THICKNESS, GAP, BIAS) \Longrightarrow (LW, FIDELITY)$. The neural network used radial basis functions in the hidden layer and a linear activation function in the output layer. The training set consisted of 125 input-output examples, sampled on a regular grid in the control parameter space (5 samples per parameter, taken in all possible combinations). Inputs and outputs were scaled between 0 and 1 to avoid ill-conditioned matrices and minimize numerical precision effects during training [23]. The NN was trained to achieve a sum-squared error of 10^{-5} or better on the training set, resulting in 112 radial basis neurons (Fig. 9). The training phase took approximately 37 seconds of CPU time on an HP 9000 workstation.

The trained NN was tested on 1327 input-output points which were not used for the training phase, and the corresponding errors were evaluated. Figure 10 show the percent error on the testing set. For the

Parameter	Sample Points
Abs. thickness	[200, 240, 300, 360, 400] nm
Gap	[10, 14, 20, 26, 30] μm
Bias	[-18, -10, 2, 14, 26] nm
Total samples	$5 \times 5 \times 5 = 125$

Table 3: *Sample points used during training.*

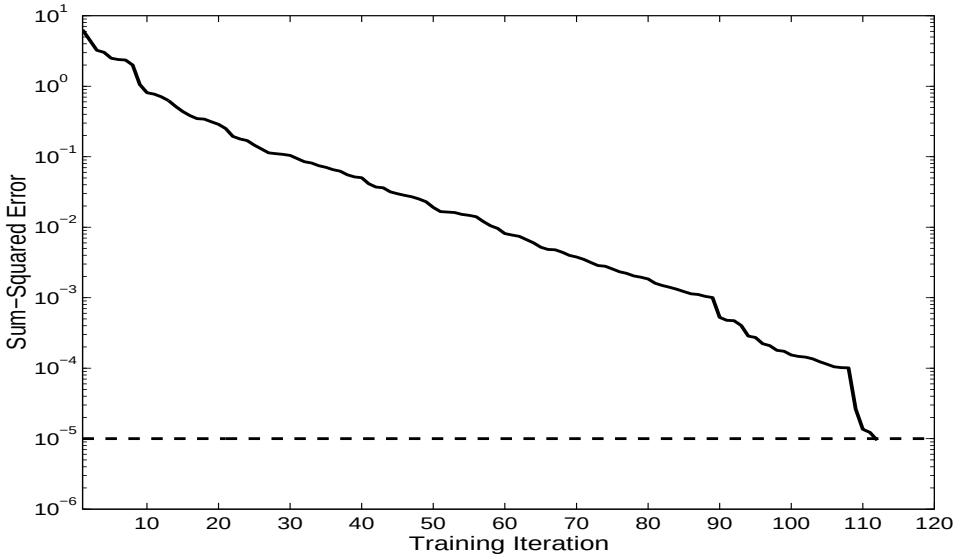

Figure 9: *Convergence of the sum-squared error toward the error goal during neural network training.*

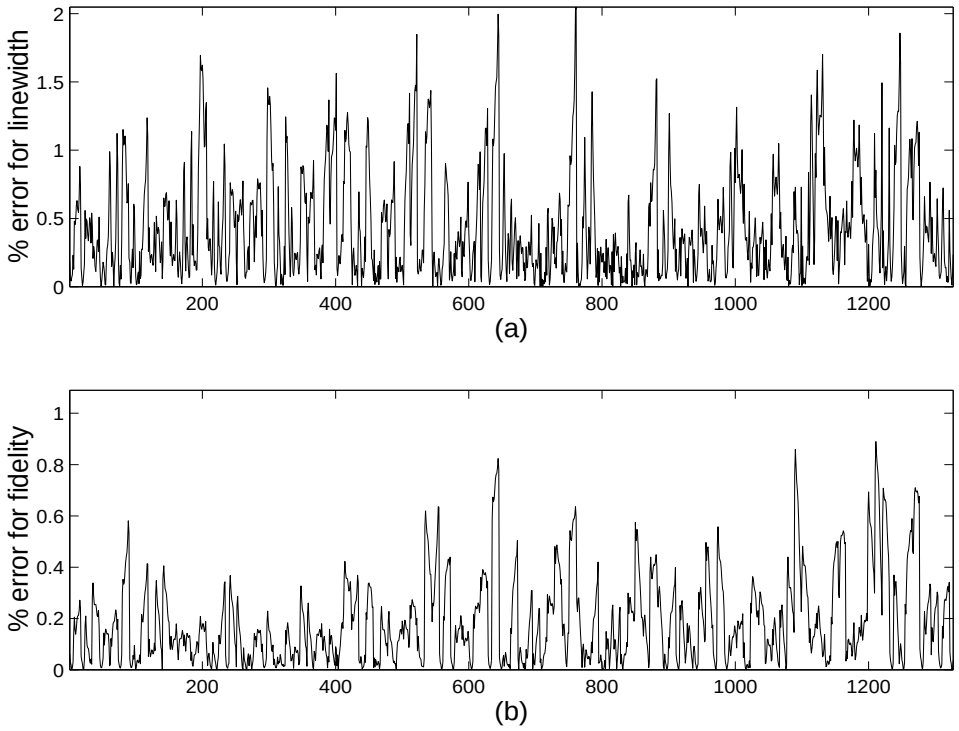

Figure 10: *Percent error of the neural network on the outputs in the testing set: (a) percent error on the linewidth and (b) percent error on the fidelity.*

LW, the error is lower than approximately 2% and for the fidelity it is below 1%. It is also instructive to observe the histogram plot of the error, as in figure 11. We notice immediately that most of the errors are clustered around zero, indicating that the NN is able to output the process responses adequately. In particular, for the LW, the median value of the error is 0.39 nm, and the maximum error is below 3 nm. For the fidelity, the median value of the error is 0.0011, and the maximum is 0.0063 (Table 4).

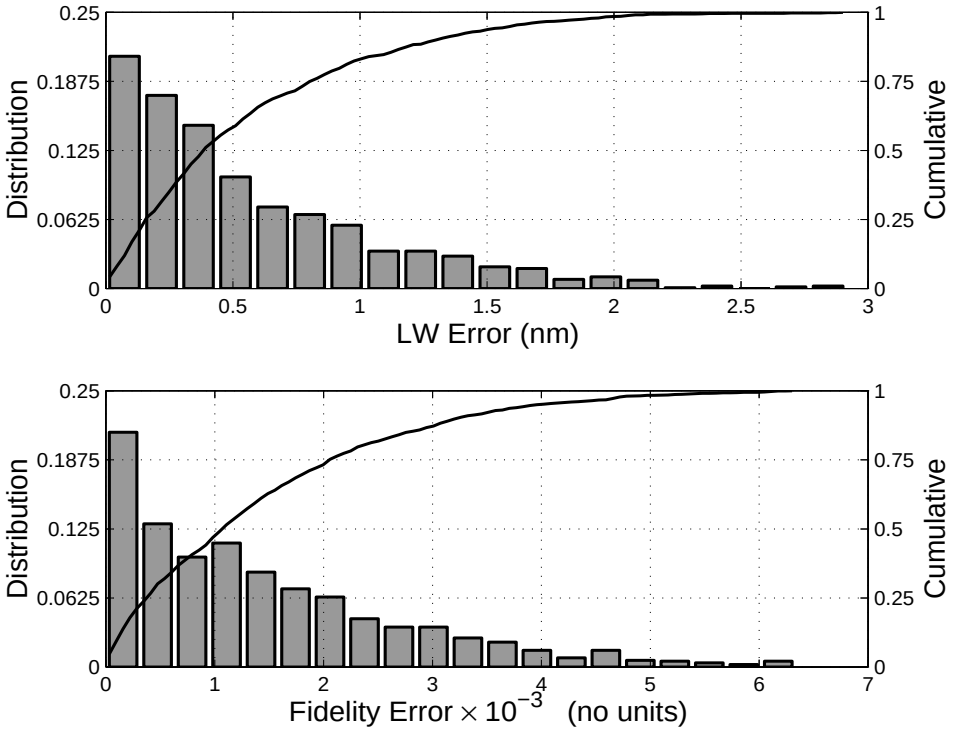

Figure 11: *Normalized distribution of the error of the neural network on the outputs in the testing set.*

	LW Error		Fidelity Error	
	%	nm	%	(no units)
mean	0.44	0.56	0.19	0.0014
maximum	2.06	2.91	0.89	0.0063
median	0.32	0.39	0.14	0.0011

Table 4: *Statistics of the magnitude of the NN error on the testing set.*

5.2 Parameter Extraction with Trained Neural Network

5.2.1 Exhaustive Search

We can use the NN model to obtain more process outputs and then perform an exhaustive search for the "best" values. The neural model was used to compute 9261 extra outputs. It took approximately 17 seconds of CPU time to carry out the simulation with the NN model, while the CXrL ToolSet would have taken several days.

The extra outputs were searched in order to find the LW closest to 130 nm with the requirement that the value of the fidelity be greater or equal to 90% of the best value (in our case greater than 0.7702, because the best value was 0.8558). The search found that the closest LW was 129.99 nm with a fidelity of 0.8297. The corresponding control parameters were (280 nm, 21 μm, -11 nm) for the absorber thickness, gap, and bias respectively.

A ToolSet simulation carried out using the values of the extracted triplet resulted in a LW/fidelity pair of (130.56 nm, 0.8261), confirming that the NN successfully learned the input-output mapping underlying the process. As further verification, the control parameters corresponding to different target LWs were extracted and the corresponding outputs computed with the ToolSet. Table 5 summarizes the results, and

Target LW	Extracted Parameters (thickness, gap, bias)	Neural Net Output	ToolSet Output
130 nm	280 nm, 21 μm, -11 nm	129.99 nm 0.8297	130.65 nm 0.8261
120 nm	280 nm, 24 μm, -5 nm	120.00 nm 0.8440	120.21 nm 0.8460
110 nm	230 nm, 15 μm, 13 nm	109.98 nm 0.7730	110.64 nm 0.7765

Table 5: *Comparison of NN output and ToolSet output for control parameters corresponding to different target LWs and fidelity within 10% of the maximum.*

again confirms that the exhaustive search strategy is successful in finding the process parameters.

5.2.2 Process Latitude

When performing parameter extraction, it is also desirable to gain insight into the *process latitude*. It is not sufficient to extract the "optimum" values of the control parameters, because in a manufacturing setting measurement noise is always present. Therefore, it may not be possible to use exactly and only the optimum values. However, we may still obtain acceptable process outputs for a *range* of values of the control parameters. The process latitude helps us understand how much freedom the process allows us in setting the control parameters while still producing desirable image patterns.

To evaluate the process latitude, we can search the 9261 data points

produced by the NN for those values which fall within an acceptable range. Typically, a LW variation of $\pm 10\%$ around a target CD is considered acceptable. In the case study we searched for process outputs within $\pm 10\%$ of 130 nm for the LW and within the top 10% for the fidelity. The reason for searching for these values is that we would like to obtain linewidths as close as possible to 130 nm, while still achieving the best possible aerial image quality.

The control parameters resulting in outputs that satisfy our search criterion define a subset of the control parameters space. Figure 12 shows a plot of the surface that includes this region. *All the control parameters triplets inside the region guarantee that the process outputs will satisfy the requirements.* This means that the acceptable region does not intersect the non-acceptable region of the parameter space.

Displaying the volume in figure 12 gives us information about the location of the acceptable parameters. We can also observe how these parameters are distributed. Figure 13 shows the bar plots of the normalized distribution of the values of the absorber thickness, gap, and bias which fall inside the volume. We can see that the values of the parameters in the "middle" of the range are more frequent than the ones closer to the extreme values. This is particularly severe for the bias. The significance of this result is that we have more latitude in choosing the absorber thickness and the gap, than the bias.

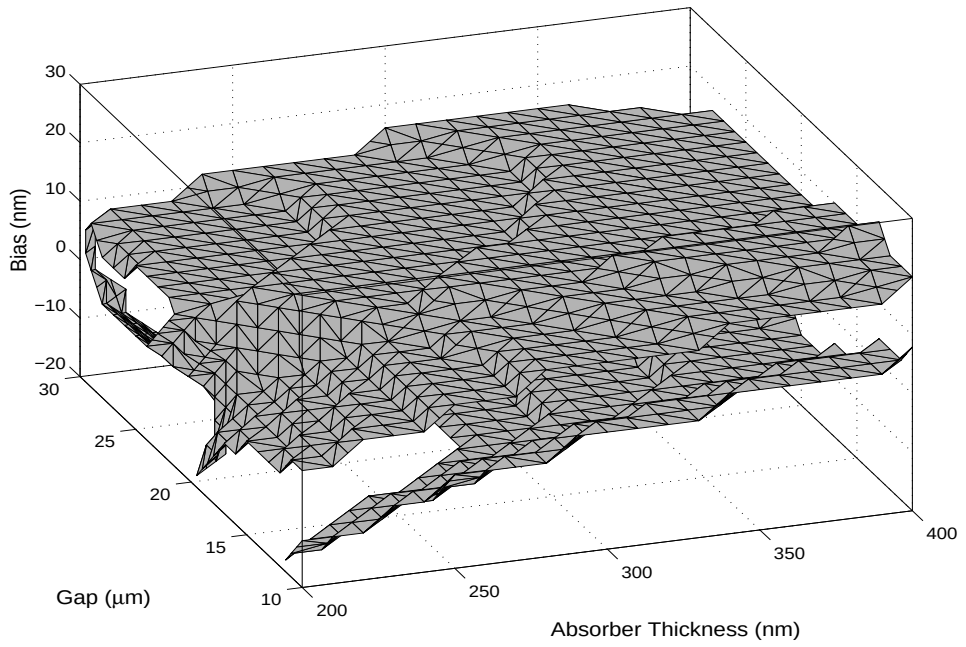

Figure 12: *The acceptable region in the control parameters space. All points inside the region delimited by the surface guarantee a LW of 130 nm ±10% and a fidelity within 10% of the maximum.*

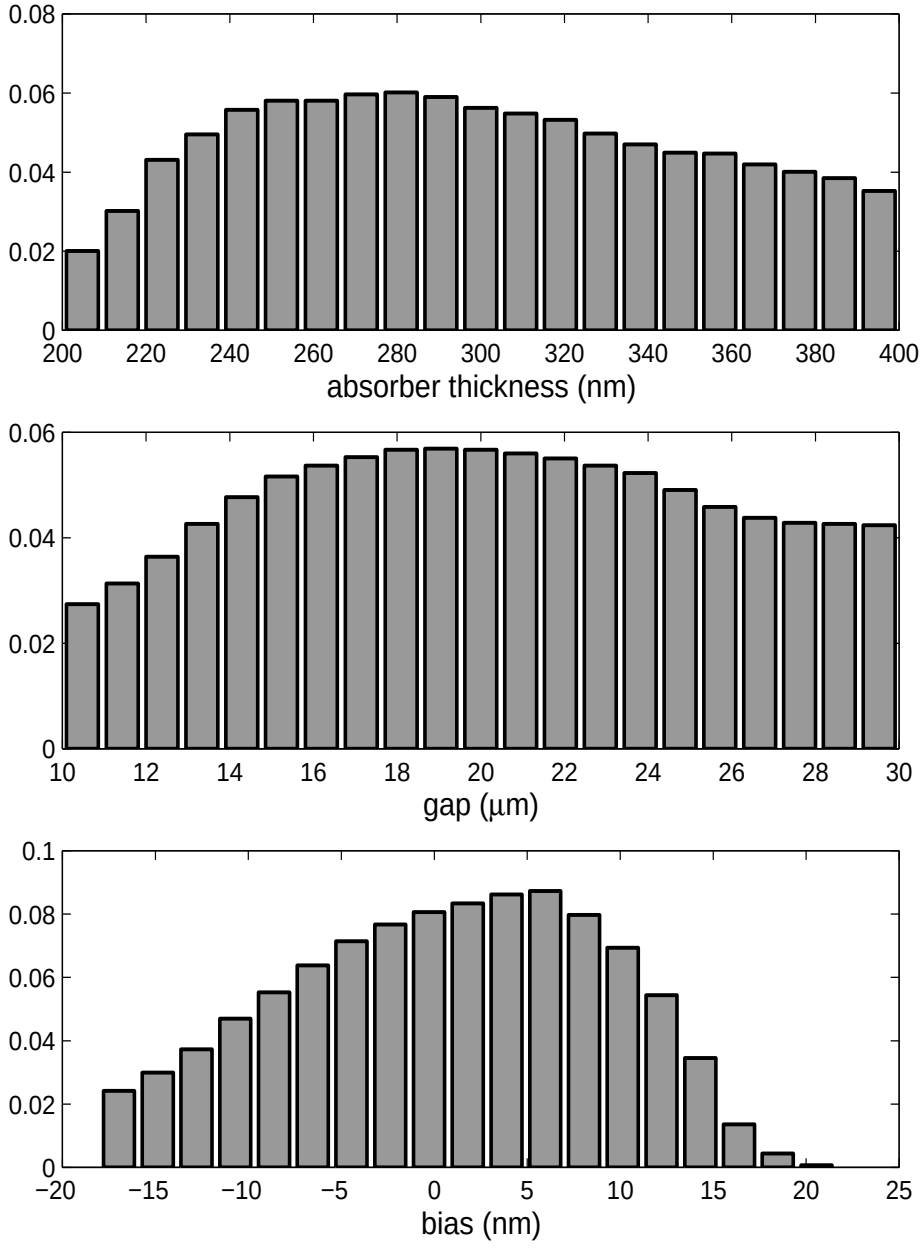

Figure 13: *Distribution of the absorber thickness, gap, and bias values which yield a LW of 130 nm ±10% and a fidelity within the top 10%.*

5.2.3 Process Latitude for Strict Conditions on the Outputs

We can further exploit the computational speed of the neural model to search for those parameters which correspond to LW/fidelity pairs that satisfy even more restrictive conditions. We can search for the parameters which yield a LW of *exactly* 130 nm (within the NN error boundaries) and restrict the choice to those which also give a fidelity value within 5% of the maximum. The following discussion describes this procedure, and serves as a verification of the speed and accuracy of the NN approach, as well as an illustration of its possible applications to the analysis of the underlying physical process.

Figure 14 displays the dependency of the linewidth on the absorber thickness and gap for a fixed bias of -11 nm. The surface was calculated by feeding the NN a very fine mesh of inputs. On the surface, the contour curve corresponding to a constant $LW \equiv 130$ nm is emphasized. This curve represents the location in the parameter space of all the $(THICKNESS, GAP)\,|_{bias=-11nm}$ pairs that will produce a LW of exactly 130 nm. Similarly, we can display the dependency of the linewidth on the absorber thickness and the bias (with fixed gap) and on the gap and bias (with fixed thickness). Figures 15 and 16 represent the resultant dependencies, for a fixed gap of 21 μm and a fixed absorber thickness of 280 nm respectively. Again, the contour curves

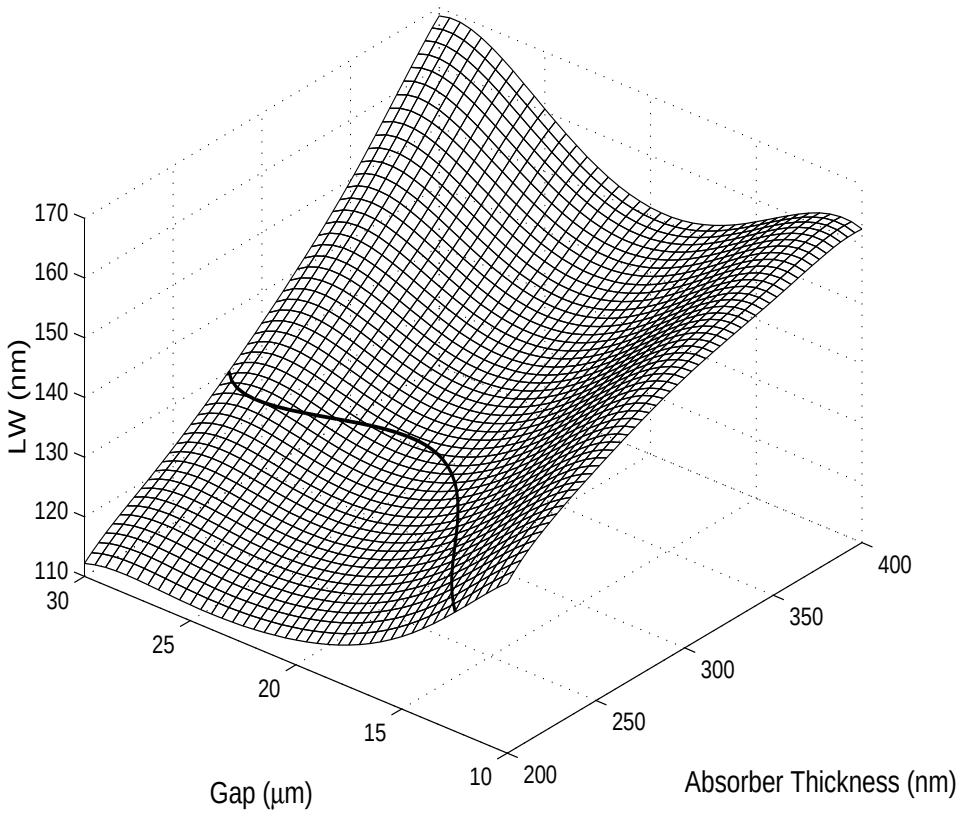

Figure 14: *Dependency of the linewidth on the absorber thickness and gap, for the case of bias = -11 nm. The dark curve represents the set of parameters which yield exactly a linewidth of 130 nm.*

corresponding to the pairs $(THICKNESS, BIAS)\ |_{gap=21\mu m}$ and to the pairs $(GAP, BIAS)\ |_{thickness=280nm}$ which result in a LW of 130 nm are highlighted by dark lines.

If we repeat this procedure for different values of the fixed thickness, gap, and bias, we obtain different curves. *All together, these curves define a surface in the 3-D parameter space. Any triplet of process parameters chosen on this surface will result in a LW of 130 nm.* Figure

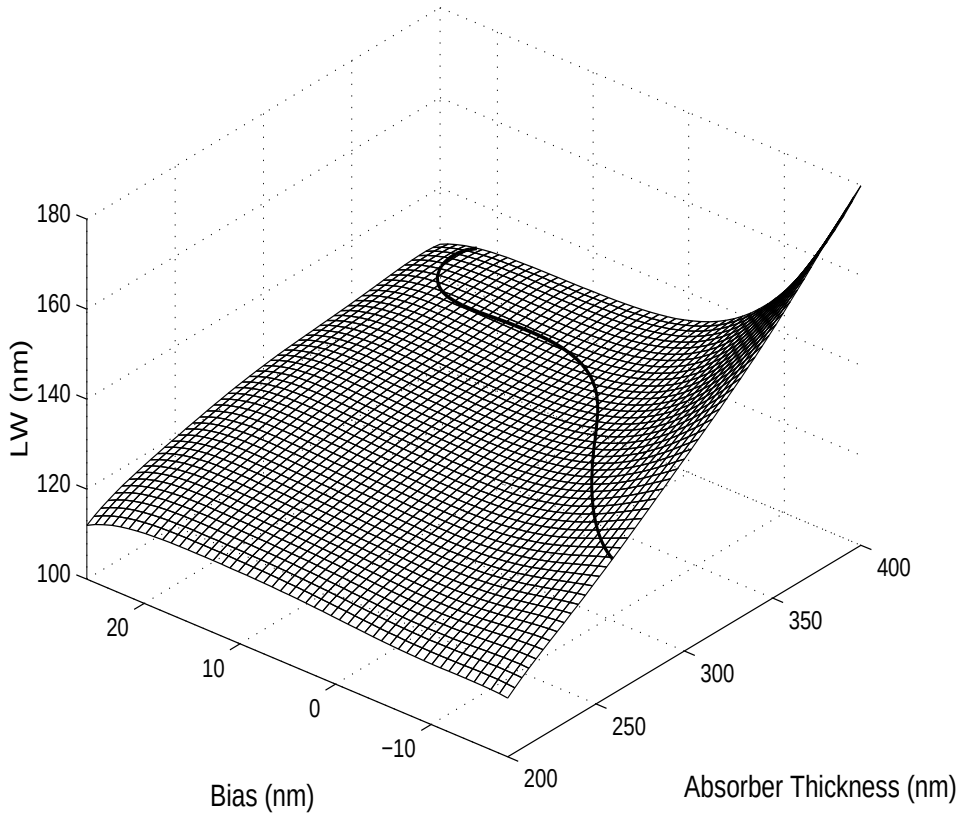

Figure 15: *Dependency of the linewidth on the absorber thickness and bias, for the case of gap = 21 μm. The dark curve represents the set of parameters which yield exactly a linewidth of 130 nm.*

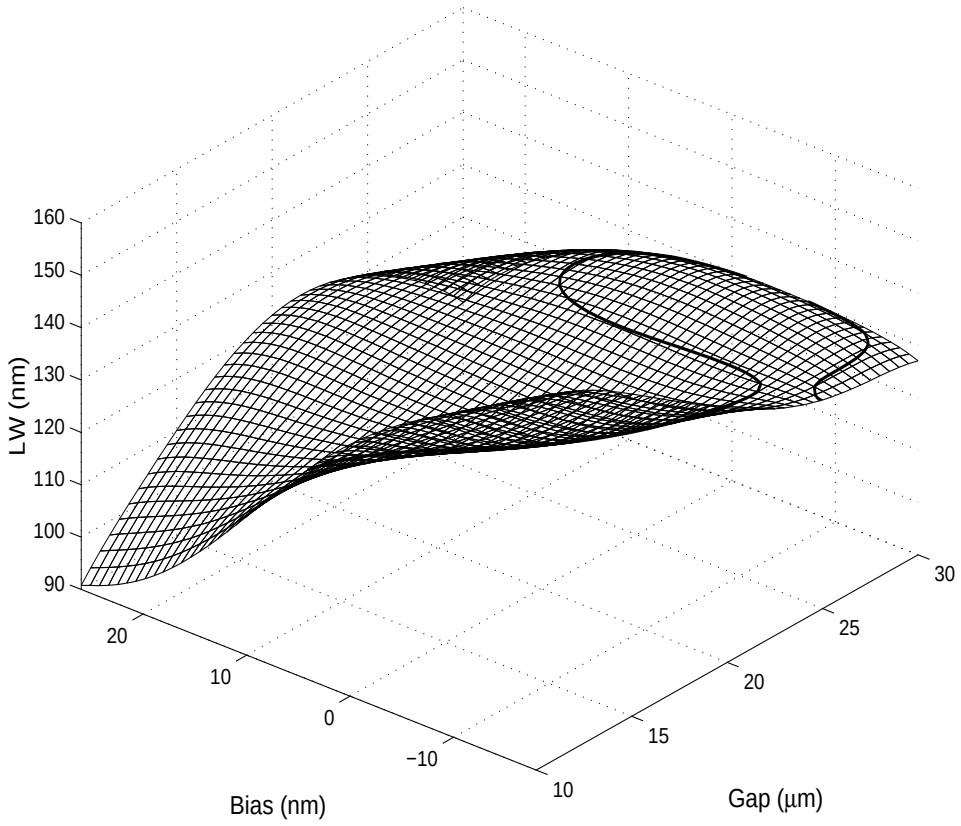

Figure 16: *Dependency of the linewidth on the gap and bias, for the case of absorber thickness = 280 nm. The dark curve represents the set of parameters which yield exactly a linewidth of 130 nm.*

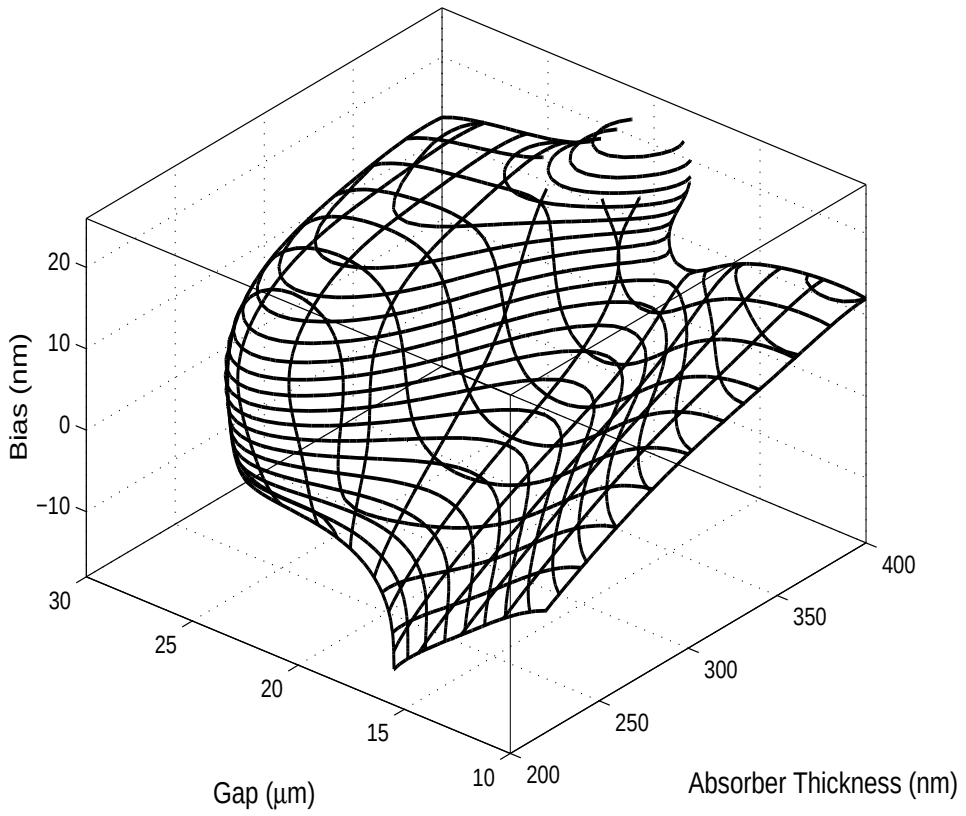

Figure 17: *The surface of constant LW* $\equiv$ 130 *nm plotted in the control parameter space. This surface was obtained by combining the contour plots of figures 14-16 for various values of the fixed control parameter.*

17 shows such a surface.

Although all the points on the surface in figure 17 will yield a linewidth of 130 nm, they might not yield a fidelity that is acceptable. Again, we can exploit the neural formulation of the process to compute many process responses and search for those values which satisfy the requirement that the fidelity be within 5% of the maximum. Conceptually, this operation is equivalent to finding the *intersection* between

the surface of constant linewidth and the volume which encloses all the parameters which yield the best fidelity values. Figure 18 shows the superposition of the surface with the "cloud" of points which defines such volume.

The points belonging to the intersection of the surface with the volume can be easily found by feeding the points on the surface of constant LW of figure 17 to the NN and then searching the fidelity output for the desired values. The end result of this search is a surface of points in the parameter space which is contained in the surface of constant LW (Fig. 19). This result shows that a *subset* of the parameters on the surface of constant LW satisfy the fidelity requirement. Figure 19 is color-coded, with lighter shades indicating a better (higher) fidelity and darker shades indicating a worse (lower) one. However, all points satisfy the minimum requirement that they yield fidelity values within 5% of the maximum.

Finally, we can observe the distribution of the newly found parameters and the corresponding process responses, as was done previously with the less restrictive conditions (Figs. 20 and 21). For the absorber thickness, we observe that the no values less than 200 nm are acceptable, and that there is a distinguishable peak at the values of 270 nm and 280 nm. Also, it is not possible to achieve the desired LW/fidelity pair with less than 13 μm for the gap. The values of the bias are even more restricted, as no values above 10 nm or below -16 nm are allowed.

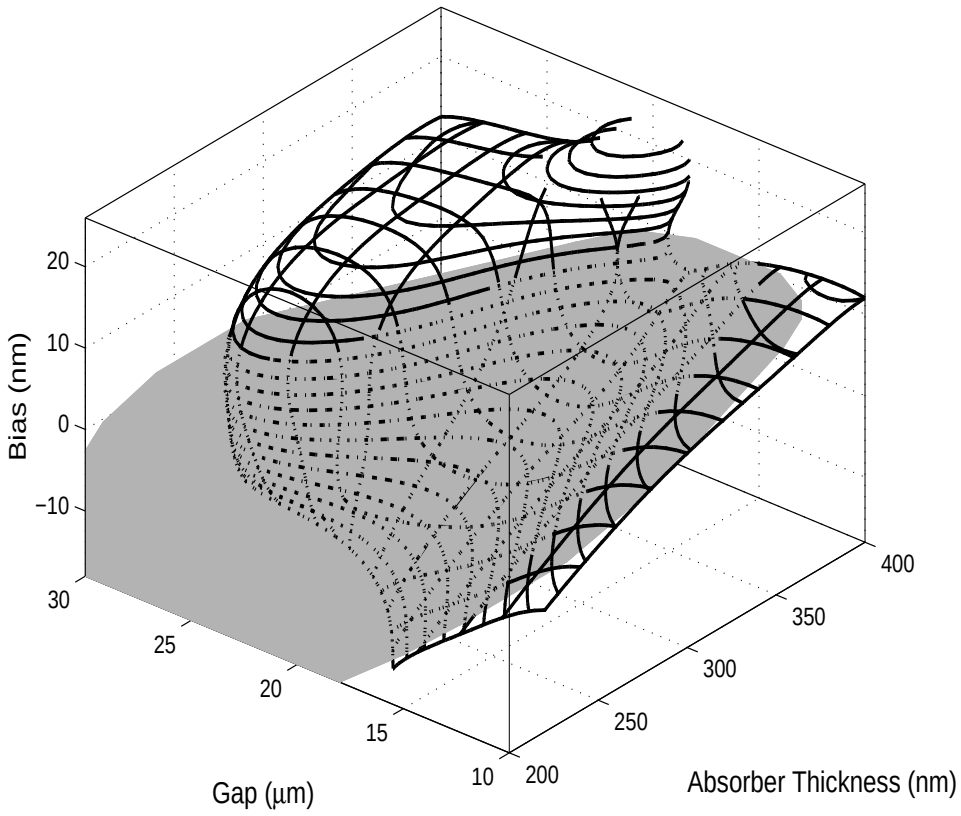

Figure 18: *The surface of constant LW ≡ 130 nm and the "cloud" of points representing the volume which encloses the parameters yielding fidelity values in the best 5%. The points that belong to both the surface and the volume (dotted part of the surface) represent the desired parameters.*

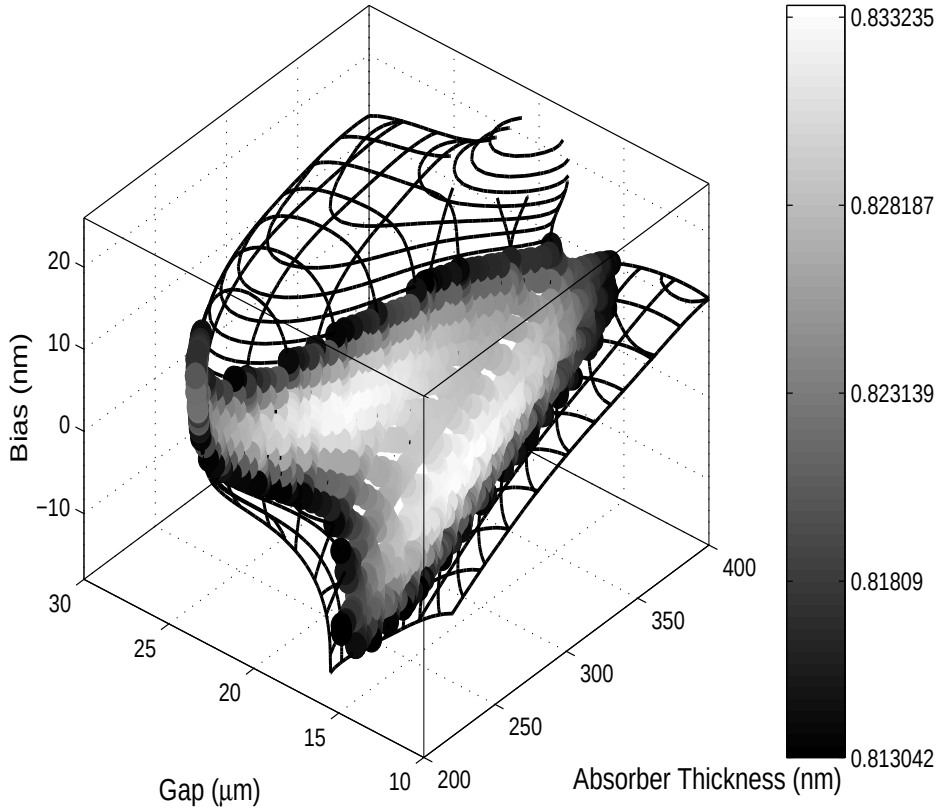

Figure 19: *Location of the parameters which yield a linewidth of 130 nm and a fidelity within 5% of the maximum value. The points superimposed on the surface of constant LW are color-coded, to indicate their corresponding values of the fidelity. Note that these points are a subset of the points belonging to the surface of constant linewidth.*

The distributions of the LW/fidelity pairs corresponding to the extracted points confirm that the LW is 130 nm (within the error of the neural net model) and that the fidelity is above the desired value. It is interesting to note that in spite of the fact that the values of the fidelity are among the best 5%, the *very best* value (0.8558 in the case study) cannot be achieved for a linewidth of 130 nm.

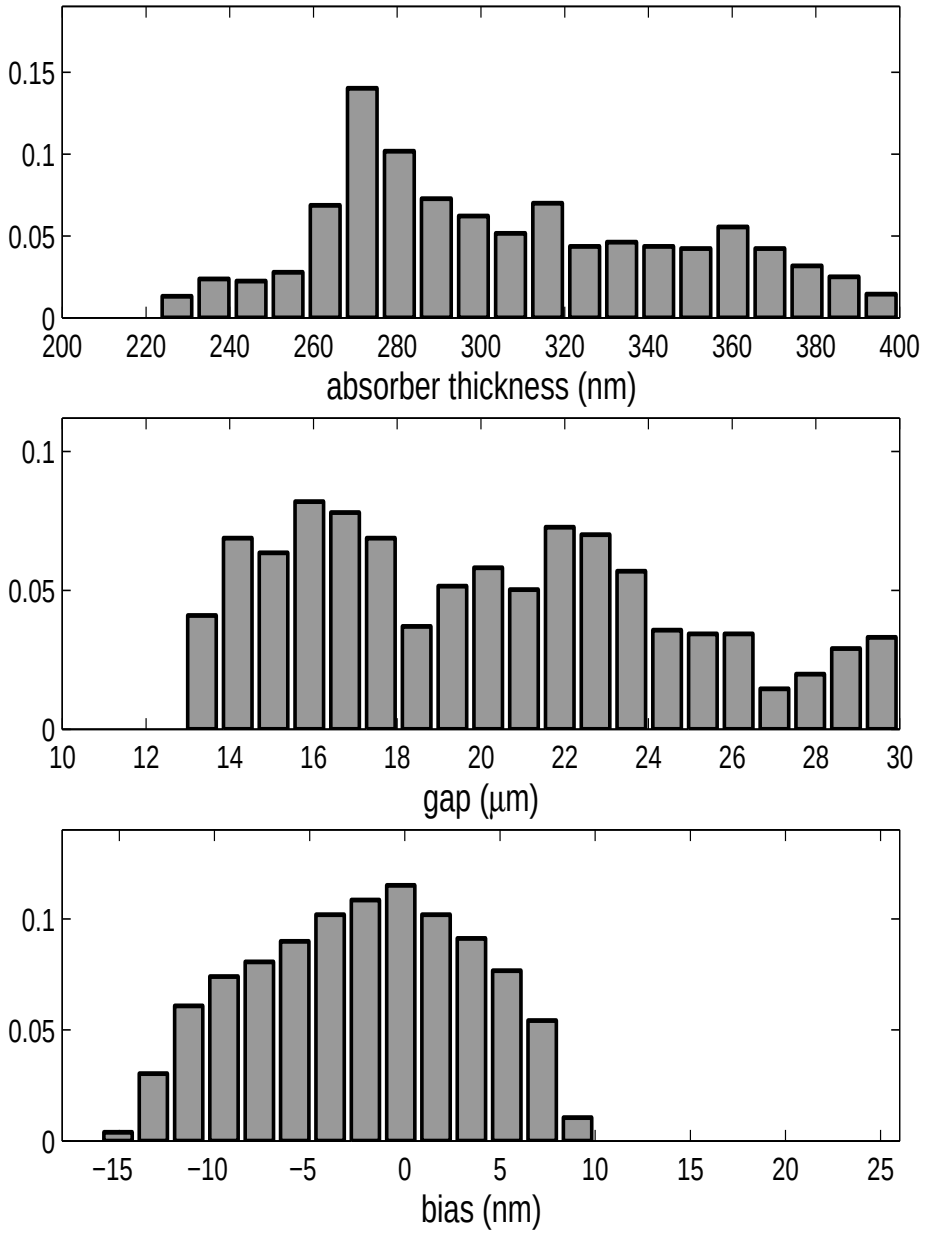

Figure 20: *Normalized distributions of the process parameters corresponding to a LW of exactly 130 nm and a fidelity within 5% of the maximum value.*

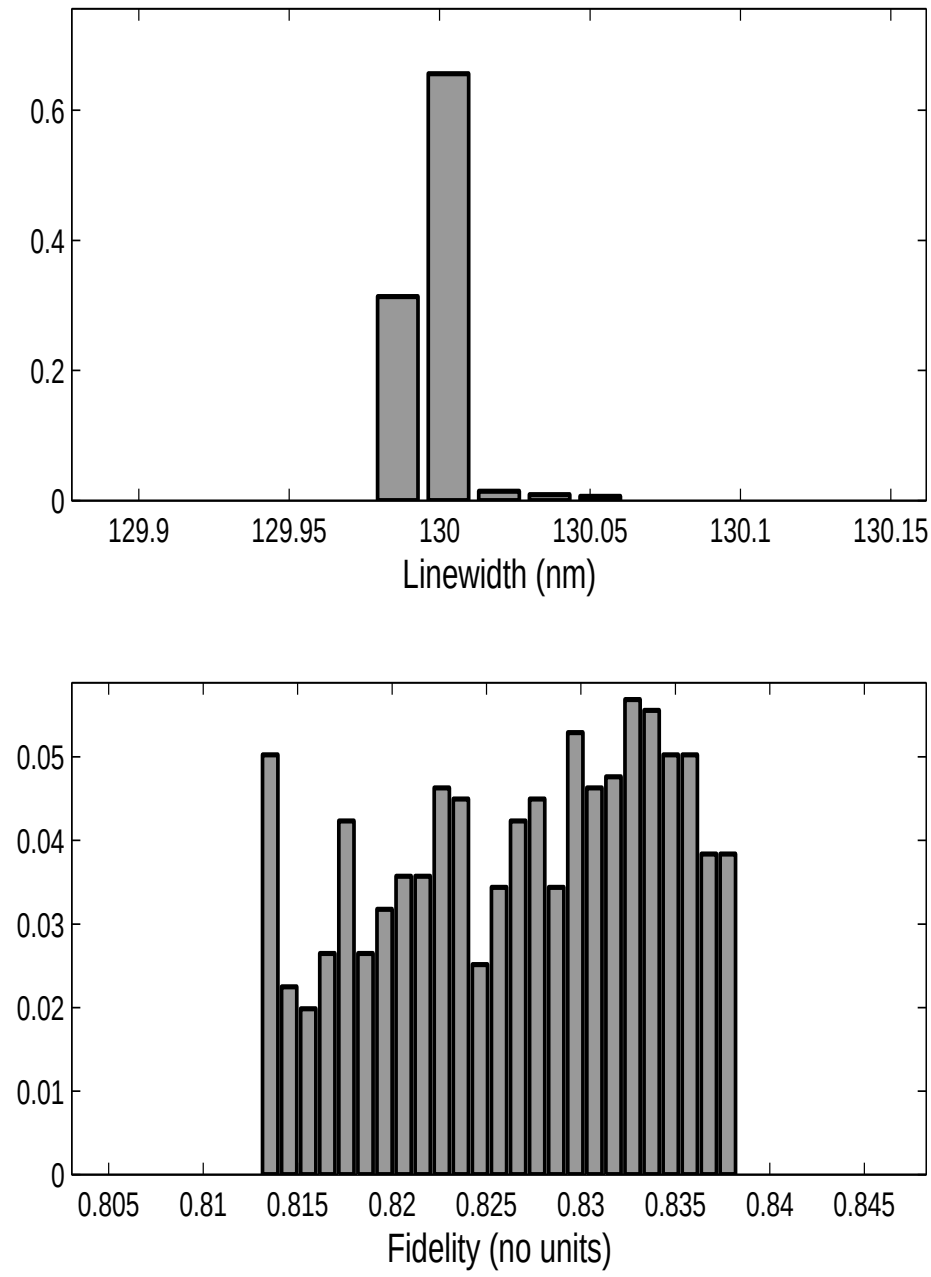

Figure 21: *Normalized distributions of the process responses (linewidth and fidelity) corresponding to the extracted parameters. These histograms confirms that the extracted parameters indeed result in desired responses.*

Chapter 6

Discussion of Results

In light of the results of the previous chapter, we can formulate a framework in which a software tool could be added to the CXrL ToolSet. We will discuss the issue of sampling the data to obtain the training set, neural network architectures, and parameter extraction procedures.

6.1 The Sampling Problem

The motivation for developing a neural model was that we needed a faster way to generate the data points necessary for an adequate analysis of the physical phenomena underlying the process. During the course of this investigation, different neural network structures were used. The backpropagation, backpropagation with momentum, Levenberg-Marquardt algorithms, and radial-basis function networks were tested. In the end, radial-basis networks resulted in the best generalization (i.e. least error).

A problem that is still an active area of research is sampling the

data. In the case study, a training set obtained by sampling the parameter space on a regular grid yielded good generalization. It is desirable to sample the neural network input variables uniformly, because we want the NN to learn the input-output relationship underlying the process equally well on the entire parameter space [25, 23]. In our case, the data is generated via computer simulations, so it is easy to specify the locations of the samples.

A limitation of using uniform sampling is that the training set grows quickly as the number of input variables increases (*curse of dimensionality*). If we have N input variables, and each is sampled uniformly at M values, then the training set will contain M^N samples. For example, in our case we had $N = 3$ input variable, with $M = 5$ samples per variable, resulting in a training set of size $T = 5^3 = 125$. If we wanted to analyze the effects of 4 variables, using the same sampling strategy would result in a training set of 625 samples. Since our motivation is to obtain data points more quickly than currently possible with the ToolSet, the advantage of using neural networks decreases as the number of variables increases. However, if the costs associated with time are of secondary importance, neural networks are the tool of choice for highly dimensional problems, in which traditional statistical techniques may be cumbersome to implement [21].

It may be possible to alleviate the *curse of dimensionality* by adopting a different strategy which can still induce the neural network to

learn the input-output mapping over the entire parameter space. Instead of sampling a few values of each parameter, and then combining all the values in all possible combinations, we could randomly sample the *vector* formed by the parameters. In this way we can still obtain 125 points, but each individual parameter will assume (approximately) 125 values, thus providing a much broader coverage of its range. In a computer simulation setting, this can be achieved easily.

Another sampling technique that has received attention is *active sampling* [26, 27, 28, 29, 30, 31, 32]. All the surveyed versions of active sampling start with a neural network trained on a small number of training samples. An algorithm then chooses the next samples adaptively, with the goal of minimizing the NN error. These techniques result in training sets which are in general smaller than the ones generated from random sampling of the parameters space.

There are two difficulties with the application of active sampling to our case study. In some cases, it is still not clear how to extend the algorithms to multi-dimensional problems. Secondly, these techniques introduce some overhead computations, and we must evaluate the cost of applying them [27]. In other words, we need to investigate whether the time spent optimizing the size of the training set (and consequently the neural network structure) will save computation time in the long run.

Radial-basis networks resulted in shorter training time and better error performance, in agreement with previous results [14, 26, 33]. While this type of NN could be trained on 125 data points in about 37 seconds on an HP 9000 workstation, backpropagation and backpropagation with the Levenberg-Marquardt algorithm did not converge or yielded poor error performance even after several hours of training.

A possible limitation of radial-basis networks is that they could result in NN structures with many neurons in the hidden layer. The reason for this behavior lies in the nature of radial-basis networks. They are superpositions of non-linear functions which learn the "local" behavior on the process, around a sample data point. Therefore, a good approximation to the "global" behavior may require the superposition of many "local" processing units [33, 34].

6.2 A Parameter Extraction Tool for the CXrL ToolSet

From the above discussion, we can conclude that a neural network combined with regular sampling of the parameters constitutes a good starting point for developing a parameter extraction methodology. We now wish to propose a framework for implementing a parameter extraction software tool in the CXrL ToolSet. This framework can be presented in several steps.

Data generation. Generate a table of input-output data points with the ToolSet. Use uniform sampling of the variables to determine the size of the training set. The training set should contain the maximum and minimum values of each parameters. In addition, generate the points to be used in the testing set. In the case study, we used 1327 points in the testing set (10 times greater than the training set), but generally 10% to 20% of the size of the training set is adequate for testing [22, 23].

NN training. Train a radial-basis neural network on the training set and verify that it generalizes well on the testing set. After a good NN is obtained, store the weights of the network in a file. These weights can be retrieved when using the NN to obtain more process points.

Parameter extraction. Use the NN model of the process to generate many more points. Store these input-output data in a table format in a file. Search the process outputs for those values that satisfy certain requirements. For example, search for the values of the fidelity that are within 5% of the maximum. This search can be repeated for other parameters, or combined with previous searches. For example, among the points with the "best" fidelity, we can search for the parameters that result in a LW which is very close to a target LW_o. The definition of target LW or any other desirable output characteristics can be defined by the user. When the searches are completed, extract the parameters corresponding to the outputs by table look-up.

Process latitude analysis. It is possible to analyze the extracted parameters to determine the process latitude and gain insight into the physical process. In the previous chapter, we illustrated how the extracted parameters lie on a surface or within a volume in the parameter space. Visualizing the results of the extraction is a useful technique that helps us understand the process. Neural networks are the tools that allow us to compute process responses quickly, and make the analysis of the results more manageable.

Chapter 7

Conclusion

The objective of the work described in this research was to explore the applications of artificial neural networks to the problem of *parameter extraction* in the framework of semiconductor manufacturing process modeling. While this approach is general, we chose X-ray lithography as a case study.

Neural networks can approximate an input-output mapping arbitrarily well, and provide a tool for analyzing highly-dimensional problems with relative ease. We have shown that a neural model of the physical process under investigation allows us to quickly compute many data points which can later be used for statistical studies and process characterization.

In the course of this research work, we were also able to provide results about the parameter extraction problem that was chosen as a case study. We showed that the process parameters that result in responses with desired characteristics belong to a surface (or volume) contained in the parameter space. The tool that allowed us to carry out this analysis in a quick and efficient way was the neural model that

we built.

We have proposed a framework which could act a basis for the development of a neural network-based parameter extraction software tool to be incorporated in the CXrL ToolSet. We have provided practical suggestions for building and training the neural network, as well as suggestions for future development. In particular, the issue of data sampling is of primary importance if we want to improve the efficiency of the approach even further. Active sampling is a technique which has received attention in the literature. However, it remains to be evaluated whether the computational overhead introduced by this technique outweighs the benefits that it introduces.

Bibliography

[1] Jerry Z. Y. Guo and F. Cerrina. Modeling X-ray proximity lithography. *IBM J. Research and Development*, 37(3):331–349, May 1993.

[2] Gary S. May. Manufacturing ICs the neural way. *IEEE Spectrum Magazine*, September 1994.

[3] Luca Cazzanti. Parameter extraction from experiment. In *Annual SRC Programs Reviews, Madison, Wisconsin*, July 1997.

[4] J. Park and I. W. Sandberg. Universal approximation using radial-basis-function networks. *Neural Computation*, 3:246–257, 1991.

[5] E. J. Hartman, J. D. Keeler, and J. M. Kowalski. Layered neural networks with gaussian hidden units as universal approximations. *Neural Computation*, 2:210–215, 1990.

[6] K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2:359–366, 1989.

[7] J. P. Card, D. L.Sniderman, and C. Klimasaukas. Dynamic neural control for a plasma etch process. *IEEE Transactions on Neural Networks*, 8(4):883–901, July 1997.

[8] B. Kim and G. S. May. An optimal neural network process model for plasma etching. *IEEE Transactions on Semiconductor Manufacturing*, 7(1):12–21, February 1994.

[9] E. A. Rietman and E. R. Lory. Use of neural networks in modeling semiconductor manufacturing processes: An example for plasma etch modeling. *IEEE Transactions on Semiconductor Manufacturing*, 6(4):343–347, November 1993.

[10] E. A. Rietman. A neural network model of a contact plasma etch process for VLSI production. *IEEE Transactions on Semiconductor Manufacturing*, 9(1):95–100, February 1996.

[11] S.-S. Han and G.S. May. Using neural networks process models to perform PECVD silicon dioxide recipe synthesis via genetic algorithms. *IEEE Transactions on Semiconductor Manufacturing*, 10(2):279–287, May 1997.

[12] Z. Nami, O. Misman, A. Erbil, and G.S. May. Semi-empirical neural network modeling of metal-organic chemical vapor deposition. *IEEE Transactions on Semiconductor Manufacturing*, 10(2):288–294, May 1997.

[13] M. D. Baker, C. D. Himmel, and G. S. May. Time series modeling of reactive ion etching using neural networks. *IEEE Transactions on Semiconductor Manufacturing*, 8 No. 1(1):62–71, February 1995.

[14] Luigi Capodieci. *Optimization Techniques for VLSI Process Modeling and TCAD in Semiconductor Manufacturing.* PhD thesis, Dept. of Electrical and Computer Engineering, University of Wisconsin-Madison, 1996.

[15] FINLE Technologies, Austin, Texas. *PROLITH/2 User's Manual*, 1995.

[16] Integrated Circuit Laboratory, Stanford University, Stanford, CA. *SUPREM User's manual.*

[17] Silvaco International, 4701 Patrick Henry Drive, Bldg. 1, Santa Clara, CA. *ATLAS User's Manual.*

[18] Silvaco International, 4701 Patrick Henry Drive, Bldg. 1, Santa Clara, CA. *ATHENA User's Manual*, 1994.

[19] B. S. Bollepalli and M. Khan. Xprocess: A brief introduction. Technical report, Center for X-ray Lithography, December 1995.

[20] C. D. Himmel and G. S. May. Advantages of plasma etch modeling using neural networks over statistical techniques. *IEEE Transactions on Semiconductor Manufacturing*, 6(2):103–111, May 1993.

[21] Simon Haykin. Neural networks expand SP's horizons. *IEEE Signal Processing Magazine*, 13(2):24–49, March 1996.

[22] Simon Haykin. *Neural Networks—A Comprehensive Foundation.* Macmillan, 1994.

[23] Kevin Swingler. *Applying Neural Networks—A Practical Guide.* Academic Press, 1996.

[24] F. Cerrina B. S. Bollepalli. On the study of aerial image quality in X-ray lithography. Technical report, University of Wisconsin–Madison, 1996.

[25] Karla Yale. Preparing the right data diet for training neural networks. *IEEE Spectrum Magazine*, 34(3):64–66, March 1997.

[26] John Moody and C. J. Darken. Fast learning in networks of locally-tuned processing units. *Neural Computation*, 1:281–294, 1989.

[27] Mark Plutowski and Halbert White. Selecting concise training sets from clean data. *IEEE Transactions on Neural Networks*, 4(2):305–318, March 1993.

[28] Vyatautas Vysniauskas, Frans C. C. Groen, and Ben J. A. Krose. The optimal number of learning samples and hidden units in function approximation with a feedforward network. Technical report, University of Amsterdam, Faculty of Computer Science and Mathematics, November 1993.

[29] David A. Cohn, Zoubin Gharamani, and Micheal I. Jordan. Active learning with statistical models. *Journal of Artificial Intelligence*, 4:129–145, March 1996.

[30] Jenq-Neng Hwang, Jai J. Choi, Seho oh, and Roberts J. Marks II. Query learning based on boundary search and gradient computation of trained multilayer perceptrons. *IEEE Transactions on Neural Networks*, 2(1):131–136, January 1991.

[31] Keiji Yamada. Learning of category boundaries based on inverse recall by multi-layer neural network. In *Proceedings of the International Conference on Neural Networks*. IEEE Press, 1991.

[32] Eric B. Baum and David Haussler. What size net gives valid generalization? *Neural Computation*, 1:151–160, 1989.

[33] Howard Demuth and Mark Beale. *Neural Network Toolbox User's Manual*. MathWorks, Inc., 24 Prime Park Way, Natick, Mass., 1994.

[34] Yu Hen Hu. Personal Communications, 1997.