SIMULTANEOUS APPROACH TO MODEL BUILDING AND PROCESS DESIGN USING EXPERIMENTAL DESIGN: APPLICATION TO CHEMICAL VAPOR DEPOSITION

A Thesis
Presented to
The Academic Faculty

by

Paul J Wissmann

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy in the
School of Chemical & Biomolecular Engineering

Georgia Institute of Technology
December 2008

SIMULTANEOUS APPROACH TO MODEL BUILDING AND PROCESS DESIGN USING EXPERIMENTAL DESIGN: APPLICATION TO CHEMICAL VAPOR DEPOSITION

The Reactor Prayer

Our reactor,

who art in my lab,

leak-free be thy piping,

thy heater work, thy computer not freeze,

during experiments, as it should work in theory.

Give us this day our daily result,

and help us quickly fix the problems,

as the problems will inevitably come against us.

Lead us not into insanity, but deliver us from equipment failure,

for thine is the hope, the promise of graduation.

Forever and ever

(or until my defense)

Amen

ACKNOWLEDGEMENTS

There are so many people to thank for helping me through this time in the graduate program, that no list can really be complete, but here goes. Dr. Ashwini Singha for his help and knowledge on anything and everything relating to CVD systems and thin film characterization techniques. Dr. Rentian Xiong for his collaboration on the CVD system and his miraculous work ethic and curiosity that inspired me to work harder and to have curiosity about everything that is going on. Dr. Cihan Oguz for being a great officemate in the dark days in the closet/office with no windows that we had to endure for the first couple of years. Dr. Jason Hicks for his willingness to give liquid nitrogen whenever I needed it, for helping me get set up in the drybox, for his expertise in organometallic chemistry, and his great friendship. Derek Jamrog, an undergraduate researcher who spent many hours with the FACET program and searching through the literature for thin film models. Sunny Shah, another undergraduate, for his excellent work on the CVD reactor to make the precursor delivery system repeatable and for his great data analysis skills. Jonathan Rawlston for being a great officemate through the years and the rest of the Grover research group for their support.

The Jones research group was always willing to lend some liquid nitrogen when I was in need and allowed me to operate in their glovebox. The Hess research group was also very helpful, especially Dr. Galit Levitin for her advice on deposition systems. All of my classmates in my graduate class who were always willing to help in any way possible, made my classes enjoyable, and made my life outside of research fun and intersting. Last but not least I'd like to thank my parents for their great support throughout the years, for always providing encouragement in no matter what I decided to do, and for always being there for me when I needed them.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Nomenclature

AFM	Atomic force microscopy
$CI_j(x)$	Confidence interval on jth model
E_d	Diffusion activation energy
E_i	Binding energy, fitted parameter for nucleation case study
$f(x)$	Objective function
F	Flux
$H(x)$	Threshold value for grid algorithm
K_{agg}	Aggregation rate [43]
K_{nuc}	Nucleation rate [43]
MSE_j	Mean square error of model j
n	Number of total experiments run
N_1	Isolated atom of size 1 ("adatom")
N_{isl}	Nucleation density on surface of substrate
p_j	Number of parameters in model j
$P(M_j)$	Probability of model j being the correct model
$\bar{x}$	True optimal point of process
$\hat{x}$	Estimated optimal point
$y(x_i)$	The i^{th} experimental data

$\hat{y}_j(x)$	Prediction from the jth model
Y	Experimental data
CVD	Chemical vapor deposition
D	Value of the discrimination function
DOE	Design of experiments
δ	Number of points used in grid algorithm
υ	Attempt frequency of adatom hopping=1e13 [1/s]
ϕ	Density
ζ	% error in $y_{j^*}(\hat{x}_{j^*})$
η	Capture number, fitted parameter for nucleation case study
θ_j	Parameter set for jth model
$\hat{\theta}_j$	Estimated parameter set for jth model
κ	Number of monolayers of deposition
ν_e	Number of repetitions of experiments
ρ	Number of adatoms needed for a stable cluster
$\hat{\sigma}_j^2$	Model variance
σ^2	Variance of the observed data
$\sigma_j^2(x)$	Prediction variance at point x
$\sigma_j^2(x)$	Prediction variance at point x
k_B	Boltzmann's constant=8.62e-5 [eV/K]

RSM	Response surface model
SZM	Structure zone model
X_j	Design matrix for model j defined in Chapter 1
Y_2O_3	Yttrium oxide
$Y(tmhd)_3$	2,2,6,6-tetramethyl-3,5-heptanedionato ($Y(C_{11}H_{19}O_2)_3$)

SUMMARY

In this thesis a tool to be used in experimental design for batch processes is presented. Specifically, this method is to aid in the development of a process model. Currently, experimental design methods are either empirical in nature which need very little understanding of the underlying phenomena and without the objective of more fundamental understanding of the process. Other methods are model based which assume the model is correct and attempt to better define the model parameters or discriminate between models.

This new paradigm for experimental design allows for process optimization and process model development to occur simultaneously. The methodology specifically evaluates multiple models as a check to evaluate whether the models are capturing the trend in the experimental data. A new tool for experimental design developed here is called the grid algorithm which is designed to constrain the experimental region to potential optimal points of the user defined objective function for the process. It accomplishes this by using the confidence interval on the objective function value. The objective function value is calculated using the model prediction of the best performing model among a set of models at the predicted optimal point.

This new experimental design methodology is tested first on simulated data. The first simulation fits a model to data generated by the modified Himmelblau function (MHF). The second simulation fits multiple models to data generated to simulate a film growth process. In both simulations the grid algorithm leads to improved prediction at the optimal point and better sampling of the region around the optimal point.

This experimental design method was then applied to an actual chemical vapor deposition system. The films were analyzed using atomic force microscopy (AFM) to find the resulting film roughness. The methodology was then applied to design experiments using models to predict roughness. The resulting experiments were designed in a region constrained by the grid algorithm and were close to the predicted optimum of the process. We found that the roughness of a thin film depended on the substrate temperature but also showed a relationship to the nucleation density of the thin film.

CHAPTER I

INTRODUCTION

1.1 Motivation

When finding the optimal settings for a batch process to create the desired amount and quality of product, one rarely has a perfect model of the process to ensure the correct settings are chosen. The optimal setting depends on the goal of the process, but is typically defined as the setting which gives the best yield, purity, or produces product at a desired speed. Whereas a continuous process normally operates at steady state, a batch process is dynamic. The state of a batch reactor is continuously changing making batch models more complex coupled with having less data available than a continous process. To find the optimal settings, one performs experiments to relate process settings to final performance and properties. The process is depicted in Figure 1. The experimental data is then combined with a model to estimate unknown model parameters. This model is then used to select additional experiments, and the model can be used to design improvements to the current process. The model for a process is chosen based on the purpose of the model, what knowledge of the process is incorporated into the model, and in what data range the model is applicable. In many cases of experimental design, an empirical model such as a polynomial fit is used [130, 97, 145, 66]. In other cases, mechanistic or hybrid models based on physical principles are used [108, 31, 122, 79]. Here, our focus will be on modeling batch processes.

An empirical model is one where a relationship between the output of a process (y) is modeled without using physical principles such as mass or energy balances. An

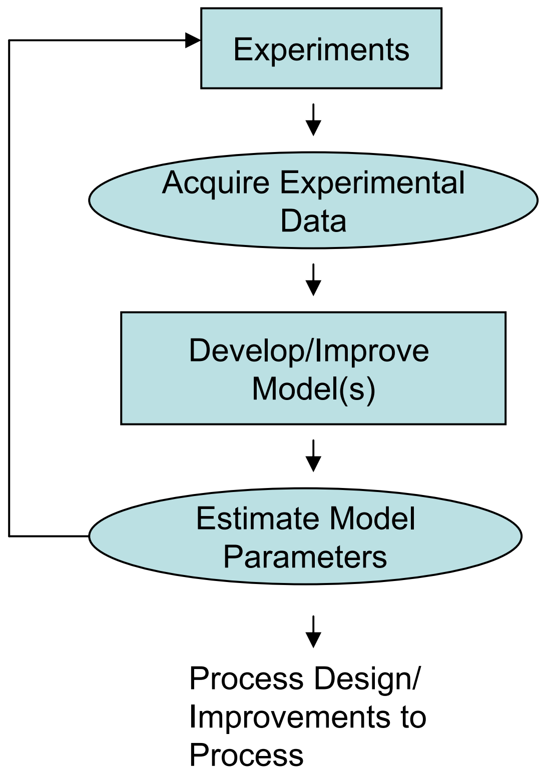

Experiments
Acquire Experimental Data
Develop/Improve Model(s)
Estimate Model Parameters
Process Design/ Improvements to Process

Figure 1: Schematic of method to find optimal settings for a process

example would be the equation of a line

$$y = \theta_1 x + \theta_2 \tag{1}$$

In this equation, θ_1 and θ_2 are parameters that relate the value of the process input x to the expected value of y. In this empirical model, the parameters do not have a physical meaning other than that θ_1 is the slope of the line and θ_2 is the intercept of the line. The values of the parameters are usually found by minimizing the error between y_{exp} (the experimental data) and $\hat{y}$ (the model's predicted value of y) using a least squares technique. An empirical model is made to fit the available data as efficiently as possible, without necessarily considering the underlying phenomena. Because of an empirical model's best-fit nature, in general it cannot be used to predict the outcome of an experiment done outside of the sample data used to fit the model.

Empirical models are commonly used to model processes that are not well understood [56], such as how many calories one can expect to burn after a certain number hours of exercising. One only wants an approximate value of the calories burned, and one does not want to model all the processes in the body that burn calories (nor are all these processes understood well enough to yield an accurate model), so the model parameters represent an aggregate of the cell processes. One could not necessarily take the slope of the line for this model and relate it back to the exact processes occurring in the body.

A mechanistic model is a model whose form is directly related to some understanding of the process, based on first principles [56]. An example of a purely mechanistic model is the ideal gas law

$$PV = nRT \tag{2}$$

which can be derived from first principles using the kinetic theory of gases. This equation is often used to model the properties of a gas in a process, and the parts of the model have physical meaning. Pressure, volume, number of moles, and temperature are measurable quantities, and the gas constant has a well established value. A good mechanistic model can help an experimenter interpolate more accurately or even extrapolate from the data given.

Hybrid models combine aspects of both empirical and mechanistic models [56]. Parts of the model may relate back to physical properties, but the value of that property may be determined by fitting it as a parameter to experimental data. A very common example of this in chemical engineering is the Arrhenius equation

$$r = A_0 e^{-\frac{E_a}{RT}} \tag{3}$$

which models the rate of reaction. Typically data is acquired from the process at different temperatures, and the values of A_0 and E_a are determined by fitting this data. A_0 is the pre-exponential factor and corresponds to the rate of reaction in the limit of infinite temperature. E_a is the activation energy for the reaction. If a molecule does not have enough energy to overcome this activation barrier, the reaction will not take place.

Many models one encounters in practice are hybrid models. As researchers gain more knowledge of the process, they add more terms to the model or change the terms in the model to reflect their deepened understanding. The added insight does not necessarily translate into knowing exact values for these terms for every system. For this reason these values are sometimes fitted to experimental data, giving the model an empirical element. Semi-mechanistic [20] and grey-box models [133] are other terms used for hybrid models. These two studies are focused on developing models from data already acquired, not on designing experiments to acquire data which would lead to better models.

In process design, a systems engineer will often use a model that has already been developed by another researcher or engineer. Unfortunately, such models can take decades to develop. Rather than wait until a highly accurate mechanistic model has been developed, a systems engineer can use partial mechanistic models. A partial model will have limited accuracy and may still be under development, but it can still be useful in process design, especially when restricted to a limited range of process settings. The engineer will start making modifications to the process with the limited data he has. By using partial models, the engineer begins using the model to make informed modifications based onscientific insight rather than modifications based purely on past performance of the process.

With the ever-changing product industry, such as microelectronics and pharmaceuticals, one will have to modify and re-optimize a process many times over the equipment's lifetime. Not only could the process be used for different products, but the specifications for products could change frequently due to new technology or tighter government regulation. The advantage of a mechanistic model compared to an empirical fit is that it can be adapted and reused for such modified situations. Due to better understanding of the process, the time needed to perform subsequent optimizations is reduced. However, empirical models are typically simpler to generate and have fewer parameters to identify. If limited experimental data is available, they may be more useful for process optimization. We are proposing a method that uses both types of models.

In a survey paper about control in microelectronics, Edgar et. al comment that advances in modeling will be needed to meet the technical challenges for that industry in the near future [42]. Modeling of processes such as etching and deposition were mentioned specifically as needing better equipment models. The lack of models for these processes has hindered improved understanding of the mechanisms involved and prevents a smooth transition from one generation of equipment to the next. The International Technology Roadmap for Semiconductors (ITRS) has also identified as a grand challenge a "strong demand on manufacturing environments to rapidly and efficiently adapt to high-mix and low-volume product runs" [60]. With a life span for equipment around five years, the transition can be very expensive in terms of time and cost. With better process models one has the capability for better design and control of the processes. Also note that it is impossible to include all effects in a process model. At some point the experimenter needs to make a tradeoff between how much time is spent building the model and how useful this model will be. The process tools in a semiconductor manufacturing facility actively produce product only 30% of the time and lacking an effective process model, cause of the problems are not easy to find or fix [42]. Using model predictive control (MPC) on these processes is one of the future directions mentioned. Obviously before one can do MPC, one needs an accurate process model on which to base control of the process. If the process is linear, one could do system identification (empirical) to get the model. However, for a nonlinear batch process this method is not so effective. Continuous process properties are often identified using a local linear dynamic model. For a batch process, however, a local model is often inappropriate and the process is nonlinear. For these reasons MPC of batch processes are very difficult.

Much of the research in microelectronics process modeling and design is still largely empirical. For example, Li et. al. used the uniform design method in their experiments on plasma spray coatings [67]. The uniform design method is a form of factorial design and uses multiple levels for each process variable to be investigated. They then tried to fit their results to a polynomial equation using regression analysis. They predicted deposition efficiency, porosity, and microhardness. They developed three different equations for each parameter they wanted to predict and then ran one trial to see how well their equation predicted. However, the benefit of developing mechanistic versus empirical models has been realized in other areas like plasma etching [51]. In fact, in the 2007 ITRS the development of first principle models for material properties was mentioned as an area where research is needed [60].

Other engineering fields that rely heavily on batch processes are biochemical engineering and bioprocesses. While much of the research on modeling in this area uses neural networks [35, 102, 128], research in process modeling is moving toward more fundamental understanding. A more fundamental understanding of the cell at the molecular level “would enable biochemical engineers to develop their bioprocesses, biosystems, and applications . . . much more efficiently and rationally” [137]. More fundamental understanding of a process generally leads to movement away from purely empirical models towards models that incorporate some prior knowledge. Once models that explain the process fully are available, the biochemical engineer will have some of the tools needed to make decisions about their systems rationally and more efficiently.

In bioprocess engineering, some of the key variables of the process cannot be monitored directly. Instead the states are estimated using a process model which is then used in process control [115]. Sensors that do not measure a variable directly but instead infer the value of the variable from other process parameters are called soft sensors. Soft sensors need models that bridge process parameters to the desired variables. These models sometimes take the form of purely empirical relationships, but empirical models are not as good at extrapolating outside of the current data range (something that could be essential when designing a new process or for model predictive control (MPC)). A model which incorporates prior knowledge of the system or first principles would be needed to overcome this, in the form of a hybrid model or even a purely mechanistic model. Xiong et al. have developed an *in-situ* sensor to estimate the states in a thin film grown via chemical vapor deposition [148].

In control of bioprocesses, the final reactor design (especially choices on the level of factors such as mixing, shear effects, and gas mass transfer rates) plays a key role [2]. With more understanding of mixing, shear effects, or gas mass transfer rates, a better model for process design could be made or other important factors could be identified to better understand the process. All three of these examples (more detailed models of bioprocesses, soft sensors for bioprocesses, and bioprocess reactor design) show a need for more fundamental understanding of bioprocesses and hence more accurate models of the bioprocesses. The need is not for another type of model or in manipulating data, but clearly much effort is being put into identifying new models for various processes. However, no model is ever perfect, so there is a need for a better method to go about finding a process model that is useful for engineering.

1.2 Process Design

Process design and process optimization are terms typically used in the chemical engineering field that mean designing a process to maximize profit [100, 125]. Much of the research in this area is focused on tradeoffs between the different setpoints that a process may have and how that affects the other unit operations in the process.

This is important when one is designing a new plant or making modifications to a plant and is accomplished using commercial software like Aspen Plus or CHEMCAD. The process sometimes already exists or the process is understood well enough to use a process model in the design software.

In contrast, this thesis is focused on process design in the sense of designing a process model to be developed and to be used during the design process. Once developed this model could also be incorporated in design software. Before one gets to the step of optimizing the whole process or plant, one must first establish the process models to use. Typical unit operations in the chemical industry such as distillation columns or filtration systems have been used for years and thus have process models that enable a user of the design software to decide on the size and type of unit operation to include in the overall process. But what about other industries such as microelectronics or bioprocess industry with new processes which are continually evolving? Traditional process design cannot really be attempted until reliable process models can be found.

Of the microelectronics unit operations, chemical vapor deposition (CVD) is one of the more complex processes. Through proper equipment design and operation one can control the phenomena occurring inside the equipment [118]. Better process models could lead to improvements in the equipment design, and also further enhance the understanding of the process. For CVD, the approaches for improved process diagnostics need to be *in situ*, due to the environment in the deposition chamber such as low pressure, substrate and film purity, and chemical precursors used [117]. With improved process models, one can develop soft sensors which can be used to better control the process [42]. For example, a reliable optical model led to the development of *in-situ* sensors to better control a chemical vapor deposition process [148].

Chemometrics is a term used by chemists that refer to using mathematics, especially modeling and statistics, to infer information from measured data and getting it into a form one can use [147]. In Wold et al.'s review of chemometrics, he points out some of the future directions that chemometrics needs to take to progress in its usefulness in industry and in academia. One point is that the methodology of experimental design is under-represented, where data analysis has been the main focus in the field for some time. Another is that empirical and mechanistic models do not need to be competing processes, but are rather complementary modeling techniques. Also, experiments are demanding more and more resources as experiments collect more data using more techniques and hence making experiments more expensive. This necessitates the use of statistical experimental design to make experiments more efficient.

1.3 Design of Experiments

Design of experiments is a method used to plan experiments to gain the most information possible from the experiments. Rippin outlined some of the basic questions that are asked while planning a new experiment [109].

1. On which subsystem should attention be concentrated?
2. What experimental configuration should be used?
3. What responses should be measured, where and when?
4. What experimental variables should be manipulated?
5. What settings should be chosen for manipulated variables?

While experimental design can help answer all five of these questions, most of the methods are focused on questions 4 and 5. To say whether the experiments are/were appropriate, some measure of effectiveness is needed. Measures such as goodness of fit and mean squared error are typically used. It is important to assess the contribution of individual model improvement to the performance of the whole system and care should be taken to use a realistic criterion for the significance of the remaining uncertainty.

Most behavior criterion during a series of experiments act the same: initial rapid improvement, possibly oscillating around a certain point, followed by a further region of steady improvement which slows down as the number of experiments is increased. After some period of time, more experiments will not improve the model and either new parameters are needed or a new model is needed if further accuracy is to be achieved. There are a wide range of experimental design techniques in research, and some of the techniques are introduced below. The research in experimental design is largely separated depending on whether the model to be developed is empirical or mechanistic.

1.3.1 Traditional Design of Experiments

When constructing models, especially the structure zone model, an empirical model mentioned in section 1.5, the experimenters were required to do hundreds of experiments in order to validate their models. When one incorporates a design of experiment (DOE) approach, it is possible to quantify more characteristics of the model with fewer experiments, using a statistical approach. DOE is generally used for empirical models. One runs enough experiments to obtain statistically significant empirical parameters for a model. Common methods used are 2^k factorial approach (where k is the number of factors or process variables in the experiment) , as well as the fractional factorial DOE [84, 140]. Disadvantages of the fractional factorial are less precision in the model parameter estimates, and confounding or masking of main effects with interaction effects. Factorial experiments are commonly used in research [11, 14, 25, 73].

Response surface modeling (RSM) is a method to predict the local shape of the response surface of a system [84]. It is used mainly for optimizing system settings and to make a system more robust. RSM is most useful when the system does not have a linear response between the high and low levels of a factor, i.e. the center point result does not equal the average of the results with high and low settings. To this end there are several approaches to DOE. Most popular is the face-centered central composite design (CCF). A central composite design is a factorial or fractional factorial experiment with center points and a group of "star points" to allow for estimation of nonlinearity. Another design is the Box-Behnken design. Unlike the CCF, this design preserves rotability but the estimation of points on the corners of the box are poor. RSM is used in many current research projects for modeling of batch processes [57, 107].

Another popular experimental design technique is the Taguchi method. The Taguchi method is best when one is trying to find the most robust operating point for a process [99]. Again, the focus here is on the result (i.e. a more robust process) rather than the knowledge about the process gained from the experiments. This method has found use in batch process modeling as well [83].

Other experimental design techniques have been used to create a better sampling scheme. Defining a regular grid on the experimental space and randomly picking points from that grid is called Latin Hypercube sampling (LHS) [40]. Alternatively, one can space the grid points irregularly based on spatial variation of the function or adaptively based on previous samples and an experimental design objective [120]. All of these sampling methods are designed for better sampling of the entire experimental region, whereas here we are interested in designing our experiments for best prediction at the unknown optimal point of a process.

Work that combines experimental design with mechanistic and empirical models has been largely limited to studies for speeding up simulation times. Specifically, the concept of surrogate models has been introduced to replace complex mechanistic models with simpler empirical models [105, 143].

1.3.2 Alphabetic Optimality Experimental Design

In the 1980s a revolution swept the experimental design community with the use of computers. Optimal design theory originated with the work of Kiefer in 1959 [88], but didn't become practical until computer algorithms were developed. The algorithms allowed "best" designs to be generated based on the experimenter's choice of model, sample size, constraints on variables, and other constraints. These criteria are characterized by letters of the alphabet and are called alphabetic optimality criteria, with the most well-known being D-optimality. D-optimality works by choosing an experimental design to achieve certain properties in the moment matrix

$$\mathbf{M} \quad = \quad \frac{X_j' X_j}{n} \tag{4}$$

where X_j is the design matrix for a model defined as

$$\mathbf{X}_j = \begin{pmatrix} a_{11}^{(j)} & a_{21}^{(j)} & \dots & a_{p_j 1}^{(j)} \\ a_{12}^{(j)} & a_{22}^{(j)} & \dots & a_{p_j 2}^{(j)} \\ \vdots & & & \\ a_{1n}^{(j)} & a_{2n}^{(j)} & \dots & a_{p_j n}^{(j)} \end{pmatrix} \tag{5}$$

and where

$$a_{lr}^{(j)} = [\partial \hat{y}_j(\theta_j, \mathbf{x}_r)/\partial \theta_{jl}]_{\theta_j = \hat{\theta}_j} \qquad l = 1, \dots, p_j \tag{6}$$

Here, $\mathbf{a}_r^{(j)} = \left[a_{1r}, a_{2r}, \dots, a_{p_j r}\right]$ is for the rth experimental setting and $\theta_j = \left[\theta_{j1}, \theta_{j2}, \dots, \theta_{jp_j}\right]$ is a vector of parameters for the jth model. The inverse of $\mathbf{M}$ contains the variance and covariance of the regression coefficients scaled by $\frac{n}{\sigma^2}$. n is the number of data points from experiments and σ^2 is the variance of the data collected. Therefore, the variance and covariance of the parameters can be controlled by controlling the moment matrix. The determinant of $X_j'X_j$ is inversely proportional to the square of the volume of the confidence region on the regression coefficients, θ. How well the set of coefficients are estimated is reflected by the volume of the confidence region. A large $|\mathbf{M}'\mathbf{M}^{-1}|$ implies poor estimation of θ in the model.

Much research has focused on D-optimal design and its applications [39, 48, 108].Other alphabetic optimality criterion are the A-optimality, G-optimality, and V-optimality [84]. The A-optimality tries to minimize the trace of $|X_j'X_j|$, which minimizes the variance on the regression coefficients of the model. G- and V-optimal designs are concerned with the prediction of the response and use prediction variance criteria. A G-optimal design minimizes the maximum scaled prediction variance over the entire design region. The V-criterion uses a set of points in the region and minimizes the average prediction variance over this set of points. D$_s$-optimal is a modification to D-optimal, which minimizes $|(A'A)^{-1}|$, using a submatrix of A composed of the set of s coefficients of interest in the model, where s ¡ total number of coefficients in the model [124].

D-optimality has been the focus of recent research in the engineering field. Kuhfeld et al. proposed a program that designs D-optimal experiments for large factorial experiments. The purpose of the program is to “free experimenters from worrying about how an array is constructed, and instead allows them to concentrate on how it will be used.” “For any researcher, finding orthogonal arrays, nearly-orthogonal arrays, and D-efficient designs can require a great deal of trial and error with different approaches, calling for many different types of expertise, none of which has anything to do with their research per se” [64]. D-optimality helps researchers determine which experimental design estimates parameters with the most confidence. Work by Franceschini et al. used experimental design to elucidate the parameters of kinetic models for a biodiesel process [44]. The drawback for D-optimal designs is the ability to only consider one model. If one is characterizing a new process, an established model may not already be available.

1.3.3 DOE and Chemical Vapor Deposition

Applying DOE to CVD experiments is not in itself a new idea. CVD experiments are very time-consuming, so the need for intelligent selection of experiments is critical. One can choose which design is best for finding the optimal setup for the system, as well as which is best for model building. Numerous works in the field have employed DOE to optimize the process [90, 12, 36, 67, 91, 110, 130]. Robbins et al. employed DOE on a plasma-enhanced CVD process to optimize electrical conductivity in nanocrystalline gallium doped zinc oxide. They employed a three-level central composite design (face centered cubic) to study the effects of RF power, pressure, and electrode gap on the conductivity and growth rate of the films [110].

In other work by Topol et al., a low pressure CVD system for depositing manganese-doped zinc sulfide was optimized using a DOE approach. DOE was used to optimize the zinc sulfide deposition using a screening experiment to find the most important process parameters and then a full DOE on the remaining parameters to optimize the deposition. Once the zinc sulfide was optimized, the investigators added Mn to the films to find the maximum luminance and efficiency for electroluminescence emission [130].

Young et al. characterized the microstructure of a diamond thin film made by microwave CVD [152]. In this study they do not use a form of DOE but the authors do a one-factor at a time approach. They identify the overall effect on hardness for each of the three factors they study, but they do not present a model for microhardness or make any suggestions for future experiments on those films. In this situation it is difficult for other researchers to use these results. Since every reactor is different, this sort of "reactor-specific" result is a common problem in CVD research.

Work on diamond-like amorphous carbon thin films used a DOE approach to optimize the microhardness of the films. Four process variables and their interactions were analyzed to produce the best microhardness of a film deposited by plasma-enhanced chemical vapor deposition [12]. They developed a RSM model as an empirical model for the microhardness. They stopped short of actually testing their model, and the paper does not mention any subsequent experiments to be performed.

1.4 Model Discrimination

In determining the best model for a process, there has been much research effort in model discrimination and parameter identification. A method for model discrimination developed by Box and coworkers [121] uses Bayesian probability to predict which mechanistic model is more probable, given existing experimental data. Equation (7) can be used for either mechanistic or empirical models,

$$P(M_j|Y, MSE_j) = P(M_j) \times 2^{-p_j/2} \times MSE_j^{-\nu_e/2} \tag{7}$$

where M_j is model number j, Y is the experimental data, ν_e is the number of repetitions for each data point, p_j is the number of parameters in model j, and MSE_j is the sum of squares of the model error:

$$MSE_j = \frac{\sum_{i=1}^{n}(y(x_i) - \hat{y}_j(x_i))^2}{n} \tag{8}$$

Here $y(x_i)$ is the experimental measurement, i=1,2,..n, and $\hat{y}_j(x_i)$ is the prediction from model j for experiment x_i. A model is penalized in Eqn. (7) for having more parameters, fewer repetitions of the experiment, or larger error than an alternative model. This method is used to discriminate between existing models using existing data, but is not designed as an iterative process for experimental design. A sequential experimental scheme was developed to discriminate between multiple models [47] by designing the experiments where the models in question have the largest difference in prediction

$$max_x(abs(\hat{y}_1(x) - \hat{y}_2(x))) \tag{9}$$

This idea of designing experiments for discriminating among models was originated by Box et al [18]. This work began with information theory and designed a criterion that functions with any number of possible models using prior probabilities. This discrimination function takes the form for m different models and n experimental points

$$D = \frac{1}{2}\sum_{i=1}^{m}\sum_{j=i+1}^{m} P_iP_j\left(\frac{(\sigma_i^2-\sigma_j^2)^2}{(\sigma^2+\sigma_i^2)(\sigma^2+\sigma_j^2)} + (\hat{y}_n^i - \hat{y}_n^j)^2(\frac{1}{\sigma^2+\sigma_i^2}+\frac{1}{\sigma^2+\sigma_j^2})\right) \tag{10}$$

where σ^2 is the variance of the observed data, $\hat{y}_n^i$ is the predicted value for the system y from model i using the first $n-1$ observations, and σ_i^2 is the prediction variance of model i. The prediction variance is a measure of the uncertainty of the model

$$\sigma_i{}^2(x) = \mathbf{a^{(i)}}(\mathbf{X_i'X_i})^{-1}\mathbf{a^{(i)}}'\sigma^2 \tag{11}$$

By choosing the next experiment that maximizes D, one attains the maximum expected discrimination among the m models. Experimental points that maximize the models' prediction differences, however, are not necessarily desirable when one is trying to gain information about the optimal design point of the process. For example, the models might actually agree at the optimal design point.

Takors et al. use a D-optimal experimental design to design experiments for model discrimination [126]. D-optimal designs are very model dependent, and if the correct model is not known beforehand, D-optimal can be troublesome. To get around this problem, the authors used a "true" model to design the D-optimal experiments. They then discriminated between 10 other models to see which model was the best out of the 10. This experimental design method is therefore dependent on having a correct model at the beginning, which is often not the case.

Previous model discrimination work largely depends on either having a relatively good model to begin with as it is often assumed one model is the true model [18, 24], or else the point where two models diverge is the best next experimental point. The work on this thesis develops a sequential experimental design technique in which one does not need to know the correct model beforehand. The method also aids in discriminating between competing models to find the best model for the process and process objectives.

1.5 Modeling of Thin Film Microstructure

In today's world of ever-shrinking length scales, thin film technology has become increasingly important and has found widespread use in a variety of applications. As device sizes become smaller, relevant length scales shrink into the nanometer range. In this range microstructure that was ignored on the bulk scale becomes increasingly important for the emerging nanotechnology. Recently there have been links made between microstructure and desirable properties of thin films in applications such as solar cells [65], microelectromechanical (MEMS) devices [23], and dynamic random access memory [141]. The properties of thin films have been found to vary according to preparation conditions [80]. For this reason a relation between material selection, preparation, structure, and property has been sought. "A good understanding could provide the basis for materials selection, process selection, and process design to tailor microstructures for optimized performance of polycrystalline films in their wide range of specific engineering applications" [127]. The ITRS has identified the impact of physical properties of materials on the electrical, mechanical, and thermal properties as an important challenge for the future [60].

1.5.1 Structure Zone Model

One of the first models attempting to address microstructure prediction is the structure zone model (SZM), first published by Movchan et al. in 1969. Their model used the reduced temperature $\frac{T}{T_m}$ to predict the final microstructure, where T_m is the melting temperature of the deposited substance [85]. The model was modified by Thornton in 1977 to include the effects of deposition pressure as well as reduced temperature. He accomplished this by adding a "zone T" for the region $\frac{T}{T_m} \leq 0.5$. The SZM has a few key assumptions [129]:

1. Incident atoms transfer kinetic energy to the crystal lattice to become adatoms
2. Adatoms diffuse over the surface until desorbed or stay in low-energy lattice site
3. The following 4 processes are significant for the process
    (a) Shadowing (which is seen via roughness measurements)
    (b) Surface diffusion: quantified by finding surface diffusion activation energy
    (c) Bulk diffusion: quantified by finding bulk diffusion activation energy
    (d) Desorption: quantified by finding sublimation energy

Notable work was done by Messier et al. where the SZM was modified to include the nanostructure, and the zone T became a subzone of the first zone, as seen in Figure 2 where the pressure is measured in microTorr [129]. The SZM was originally developed to predict a sputtering process, but has since been adapted to include deposition processes such as CVD and sol-gel deposition techniques. Schuler et al. were able to predict the microstructure of a thin film grown via a sol–gel technique by using a ratio of the thickness of a single atomic layer and the size of the crystal grains. They extended their model to CVD by considering the layer deposited to be infinitesimally small. When one makes this assumption their model agrees with that predicted by SZM [116]. The SZM model was developed under the assumption that temperature and pressure remain constant throughout the deposition which could be a significant limitation when the model is used for process design. The SZM model is an example of a qualitative mechanistic model. After doing many experiments, the researchers were able to describe mechanisitically (what process dominated the deposition and

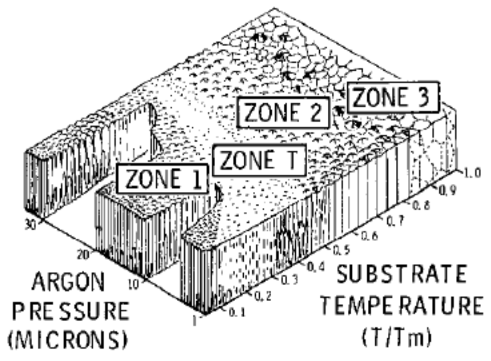


Figure 2: Structure Zone Model [129]

microstructural formation) what ocurred on the surface and qualitatively described the resulting microstructure.

The SZM model has also been used as the basis for computer simulations of thin film growth. Savaloni et al. developed a 2D model to predict microstructure by constructing an algorithm which takes into account the mobility of the atoms once they reach the surface of the film. The model calculated the probabilities of an adatom hopping based on the temperature of the substrate and the amount of nearest and second-nearest neighbors. They varied the substrate temperature, the deposition rates, and the angle of incidence and the model gave results that agreed with the SZM [114]. However, at higher deposition rates their simulation resulted in dendritic growths on the substrate which is not observed in experiments.

1.5.2 Kinetic Monte Carlo

Other attempts to model thin film microstructures have used kinetic Monte Carlo simulations (kMC). These simulations use kinetic parameters obtained via experimentation to model the film morphology. The model developed by Ni et al. is similar to the one by Savaloni mentioned earlier in that it includes adatom hopping on the substrate surface, but in 3D and atoms were allowed to hop over longer distances than in the Savaloni model. This model was used to predict the roughness of a deposited surface and was extended to films with two components. It was also used in simulations to control the roughness of a film [89]. This model used generic molecules so one cannot compare to the SZM since a reduced temperature requires a T$_m$ which is specific to a material.

Work by Rubio et al. simulated depostion and annealing of three-dimensional polycrystalline thin films to understand the role of elementary atomistic diffusion mechanisms on the deposition and thermal processing of polycrystalline thin films [113]. An event manager controlled the timing of the simulation and ran the two types of events in the simulation: deposition and diffusion. Deposition was modeled as atoms approaching the substrate at a rate equal to the desired growth rate. Diffusion was modeled by atom jumps which occur at a probability which was proportional to a rate which depends on migration energy and on the change in system energy due to the jump. The model was used to study the effects of temperature, deposition rate, and adhesion energy on the film microstructure. Monte Carlo (MC) models are typically hybrid models as the activation energies are estimated from experimental data or from molecular dynamic modeling. MC models are typically used to model uncertainties which can be used in experimental design [55, 138]. However, experimental design has not specifically been used to estimate parameters in a Monte Carlo model but this could be done. A finite difference method could be used to estimate the X_j matrix and experiments could be designed to improve a MC model.

1.5.3 Grain Growth Models

Another approach to modeling the microstructure of thin films is through studies of nucleation, grain growth and grain coalescence. The microstructural characteristics of importance include grain shapes, grain sizes, distribution of grain sizes and distribution of grain orientations [127].

A model tracking the motion of grain boundaries was developed by Zhang et al. They constructed a simulation code to predict the microstructure of metallic thin films. The program makes use of 1D and 3D kinetic lattice Monte Carlo (KLMC) models to predict the profile and microstructure (grain size, grain shape, grain orientation, texture, and roughness) of thin films as a function of their deposition conditions (temperature, flux distribution, deposition method, substrate geometry, materials). KLMC makes use of an atomic lattice to move the atoms in the simulation and is also known as atomic kinetic monte carlo. The program they developed is called FACET and is a downloadable program from their website [154, 155]. To reduce the computational time of their model, they make a few key assumptions:

1. It is a two-dimensional simulation.

2. They describe facets and grain boundaries with line segments. Each facet is described by one line segment, while boundaries are described by multiple line segments to show proper grain shape and film structure.

3. Each nucleus has its own unique orientation which affects facet growth rates. Each grain is allowed to rotate by any angle around the axis.

4. Initial size, texture, and nuclei density are input factors.

The program calculates how the facets of a film move. They can jump to another facet, jump to the substrate surface, or interact with an adjacent grain possibly combining into one grain. The user inputs the deposition flux and the activation energy for each type of diffusion. The program then calculates the jump rate or probability for each jump direction and determines the new facet location. The purpose of this program is to provide a tool for virtual experiments. The software gives the user an approximation of what the film may look like after the experiment.

Friedrich et al. developed a program called GROFILMS to model a 2D thin film microstructure [46]. This model uses line segments to simulate fundamental film growth phenomena. This model addresses issues such as substrate wetting, grain growth, crystal facetting, grain boundary grooving, shadowing, internal microstructure, and surface topography. The paper does not necessarily model a paticular film-building process like CVD or sputtering, but alludes to sputtering as the desired type of system. They compare their model to the SZM to show that their model is valid. The parameters they use to generate a film are substrate surface tension and interfacial tension. They do not use any experiments to back up their predictions, but instead compare the predictions between the two models.

Rollett et al. simulated the growth of abnormal grains using a Monte Carlo model [112]. By taking into account the different grain orientations possible in a thin film, they used the concept of grain boundary energy to simulate grain growth. Unfortunately, grain boundary energies cannot be estimated easily on a typical deposition system.

1.5.4 Nucleation Models

Some of the grain growth models need a nuclei density to proceed. Unfortunately, nuclei density are generally too small to measure experimentally, so many models have been developed to describe the nucleation of thin films as well. Zinsmeister et al. described the nucleation of a thin film using a system of stochastic equations. Each equation described the mobility of a size of cluster up to a cluster of n atoms and used fitted parameters such as a collision factor, adsorption energy E_a, and binding energy, E_b to develop their hybrid model [157]. One of the problems mentioned in modeling nucleation is the short time window of nucleation. Most analyzation techniques take a second to acquire data from a nucleating thin film which is unfortunately too long to see the developing nucleation islands [157].

Other work in nucleation modeling has focused on the three types of growth for thin films [136]. Particle growth form small clusters as the film grows. True layer growth or Frank-van der Merwe growth is characterized by each layer being identical to the previous layer and is often seen in the homoepitaxy of metals and semiconductors. Stranski-Krastenov or layer-plus-island growth has identical initial layers, but then forms islands. This type of growth is typically found in heteroepitaxy of metals and semiconductors. Stowell et al. did numerical simulations of nucleation density using deposition rate, J, adatom diffusion, D, and the binding energy of n atoms, E_n [123].

Evans et al. developed a nucleation model using "mean-field" rate equations, a set of partial differential equations that describe the nucleation process [76].

$$\frac{N_1}{dt} = F - (\rho + 1)K_{nuc} - K_{agg} \tag{12}$$

$$\frac{N_{isl}}{dt} = K_{nuc} \tag{13}$$

where F is the flux of atoms to the surface and ρ is the number of adatoms needed for a stable cluster. The rate of nucleation is calculated

$$K_{nuc} = \eta_\rho h N_1 N_\rho \tag{14}$$

where η_ρ is the capture number for a cluster of size ρ, h is the hopping frequency, N_1 is the number of adatoms on the surface, and N_ρ is the mean density of clusters of size ρ. $h = vexp(-\beta E_d)$ where $v = 1 \times 10^{13}$, the attempt frequency for hopping, and

$$\beta = \frac{1}{k_B T} \tag{15}$$

where k_B is the boltzman constant, and T is the temperature.

$$N_\rho = c_\rho exp(-\beta E_\rho)(N_1)^\rho \tag{16}$$

where c_ρ is the number of configurations of stable clusters, E_ρ is the binding energy for the cluster. The rate of aggregation of adatoms is $K_{agg} = \eta_{av} h N_1 N_{isl}$. These rate equations are also hybrid models since the energies need to be estimated from experimental data.

1.5.5 Multiscale Modeling

Modeling of processes has occurred on many different scales which usually depends on the level of understanding of the process being modeled. Continuum models on the gas flow in a CVD reactor have been done [58, 22, 95], as well as modeling on the microscale which are represented by the grain growth and nucleation models mentioned previously. The macroscale continuum models are useful for describing flow patterns within the reactor as well as analyzing the heat transfer within the reactor which definitely have an effect on the microstructure. The drawback is that such models cannot describe the microstructure of the thin films very well unless it is coupled with a microscale model.

Multiscale modeling of CVD processes has also been attempted for linking macro- and microscale processes [77]. Jensen et al. developed a model in which the microscale film growth provided boundary conditions for the macroscale model of chemical species transport. While previously described models focused on surface evolution with adatom hopping, this model simply looks at the probability of whether or not a species will actually stick to the surface or go back to the bulk gas phase. They introduce an effective reactivity function which takes into account the number and nature of encounters a molecule will have going from the bulk phase to the substrate surface. The function is calculated statistically by finding the possible trajectories a molecule could take from a point some distance from the substrate surface. Then the function is used as a boundary condition for macroscopic simulations [111]. However, this model does not describe the microstructure of the resulting film.

Yu et. al. developed a method for microstructure prediction in thermo-mechanical processing in metals [153] by trying to bridge macroscopic and mesoscopic models. The mesoscale is smaller than the macroscale of substrate diameter, but it is larger than modeling the individual atoms which would be the microscale. They attempted to combine FEM and MC simulations. Their methodology took a continuum-based model (FEM) and used that information for an empirical (MC) simulation. They presented FEM results and MC results from other papers, but they did not combine the two and only explained the methodology they are going to use. Again, this is a modeling study, with no experimental validation or demonstration. In this thesis the main focus will be on microstructural modeling, but macroscale models such as growth rate will also be included. Although the focus will be on the microstructure, the macroscale features of the reactor cannot be ignored.

1.5.6 Observations from Current Models

Looking at the current simulations available today, one sees many different approaches to modeling the surface and microstructure evolution. From empirical modeling via SZM to modeling of film evolution to kMC to multiscale modeling, all methods seem promising in the end goal of predicting and controlling thin film microstructure. Empirical models are simple to use in design but assume constant process settings and because of their simplicity have limited predictive power. Mechanistic models can provide more insight, but are more difficult to use for design. Also, they require more experiments to identify parameters or the process settings are not physically observable. One thing that is lacking is an approach attempting to combine the different modeling techniques currently in place. Finding a link between SZM (empirical models) and kMC and/or multiscale modeling could lead to a model that captures the evolution of the microstructure. We want to combine the best features of each approach and use them simulataneously to conduct the *best* experiments to build the models. This is especially important since experiments are slow and costly, and no single model is ever the "true" model anyway.

1.6 Microstructure Optimization

Microstructural optimization has also garnered attention in the research community. Medina et al. have developed a method of optimizing the microstructure of a hot metal extrusion process [81]. Here the process was well understood fundamentally and uncertainty in the model was less of a concern. A model for microstructure was used to determine the best process settings to obtain the desired microstructure. They then actually designed their process around the desired trajectory of the process. This is an example of process design that models developed by the method in this thesis would enable. Once a good model for microstructure is obtained, process optimization such as in this example becomes possible.

1.7 Chapter Outline

In Chapter 2 a full description of the metalorganic chemical vapor deposition (MOCVD) system is given. This is the system which our assumptions and theories were tested on. The design of the hardware is given, as well as some equations used in the PI control loop for precursor delivery are detailed.

In Chapter 3 a description of experimental design is given and active areas of experimental design research are described. These elements are then combined into an experimental design methodology to be used throughout the thesis. The basic methodology is shown in Figure 3.

In Chapter 4 a simulation study based on the experimental design developed in Chapter 3 is presented. The simulation focuses on refining the experimental design to be used on an experimental system. First, simulated data using the modified Himmelblau equation is used and the results of fitting a model to that data with added noise is presented. Next, a simulated study of thin film growth is presented with multiple models to fit the simulated data with added noise.

In Chapter 5 the results from Chapter 4 are used in implementing this experimental design on an experimental MOCVD reactor system. The methodology is used to design experiments to find the optimal point of operation in the process to achieve the optimal roughness of the film.

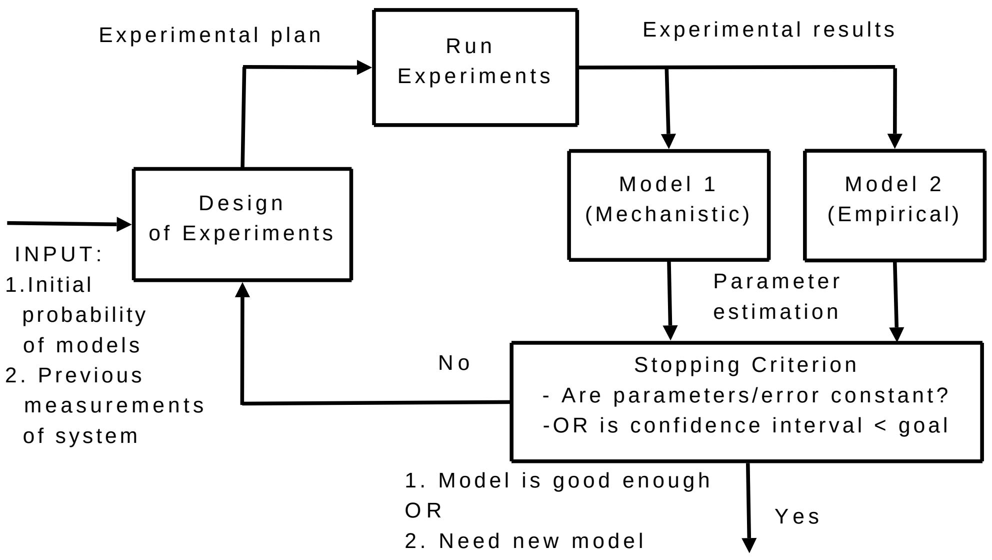


Figure 3: Proposed experimental design algorithm. Two models are shown here, but any number of models may be used.

In Chapter 6 some additional considerations for experimental design are presented. The technique of identifying a poor model is presented and some additional work on the experimental design methodology is presented in a study to fit the growth time of the CVD reactor. Also presented is ensuring whether your metric upon which to build your model is robust. An attempt to measure grain size is shown and the failure of this metric for model design is highlighted.

In Chapter 7 the conclusions of the thesis are presented, as well as future directions of research. This work has unique contributions to the field of model building, process design, and experimental design. This is the first experimental design methodology to our knowledge with the specific aim of developing process models. Mechanistic models are often developed removed from experimentation, and the result is a model that does not relate well to the actual process and cannot be applied to the process directly. Alternatively, experimental data is used to develop empirical models, but these models are only useful in the range where the data was collected and should not be used to extrapolate into other regions. By anchoring the model in experiments while also testing hypotheses for deeper understanding of the process, a practical model for experimenters can be developed.

This work also demonstrates the need for researchers with a diverse skill set. While many are either experimentalists or theorists, there is a definite need for researchers capable of working in both realms or at least capable of working closely with another researcher with a complementary skill set. Models developed without experiments will be unlikely to have a practical application, while models developed without theory will have limited predictive range.

CHAPTER II

CHEMICAL VAPOR DEPOSITION EXPERIMENTAL TESTBED

To test our theories and assumptions an experimental testbed was needed to provide experimental data. Chemical vapor deposition (CVD) is a common batch method for depositing thin films and is used in a variety of applications such as microelectronics [52, 86, 141], thermal barrier coatings [135, 139, 156], and fuel cells [72, 82, 146]. The strength of CVD is the ability to deposit uniform films over non-uniform surfaces, as well as the advantage of not requiring high vacuum or needing line-of-site to the substrate for deposition, as physical vapor deposition methods require [117]. The exact mechanisms occuring during CVD are not always very well understood or quantified, and for this reason on the factory level many substrates are wasted trying to find the correct recipe to deposit the desired microstructure [42]. If one had a better understanding of the underlying mechanisms, one could design better CVD systems to deposit the desired mechanisms. However, due to the quickly changing nature of the microelectronics industry, always making devices smaller and smaller, a detailed study of these mechanisms is difficult. For this reason, a low pressure metalorganic cold wall chemical vapor deposition (MOCVD) system was constructed to test the experimental design methods developed. This system was originally constructed to deposit yttria-stabilized zirconia thin films, but for simplicity we currently deposit only yttria oxide (Y_2O_3).

Thermal chemical vapor deposition works by heating a substrate to high temperatures and flowing over the substrate the reactive gases which contain the chemical to be deposited. Metalorganic precursors are used in CVD due to their low sublimation

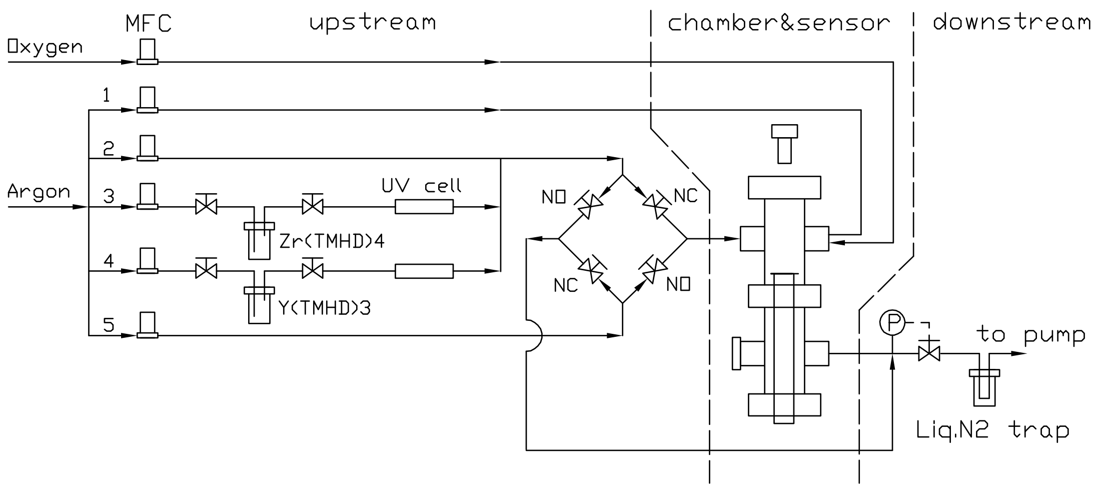


Figure 4: Schematic of the chemical vapor deposition system showing the three sections: upstream, chamber and sensor, and downstram. "NO" and "NC" denote normally open and normally closed valves respectively. "P" denotes the pressure sensor and "MFC" denotes mass flow controller.

temperatures and highly reactive nature [104, 103]. At high temperatures ($> 600°C$), the precursor compound begins to decompose and the metal deposits on the surface creating a thin film. The reactor system can be divided into three parts: upstream, deposition chamber, and downstream. This is shown schematically in Figure 4.

2.1 Upstream

Figure 5 shows a photograph of the experimental testbed in our laboratory. The upstream portion consists of six mass flow controllers (MFC) [MKS Instruments], two precursor evaporators [Kurt J. Lesker], two UV cells [Ocean Optics], a flow manifold which is described later, and stainless steel tubing connecting all of the VCR fittings [Swagelok].

The evaporators are cylindrical containers of stainless steel. Each container is 5" tall and 2" in diameter covered with a 3-$\frac{3}{8}$" Conflat vacuum flange. The cover has three tubes going into it: a $\frac{1}{8}$" sealed tube for a K-type thermocouple, an inlet tube ($\frac{1}{4}$" OD tubing) that extends 4-$\frac{5}{8}$" into the container, and an outlet tube ($\frac{1}{2}$" OD tubing). The inlet tube introduces Argon gas into the evaporator to dilute and help carry the precursor to the rest of the reactor. A larger tube diameter is chosen to carry the precursor vapors to prevent clogging in the tubing due to condensed or decomposed precursor and $\frac{1}{2}$" tubing is used whenever the gases contain the precursor vapors. Both the inlet and the outlet tubing have a manual valve on them to either open up or close off the evaporator from the rest of the upstream system. An evaporator is packed with precursor inside a drybox [Unilab] by grinding up the yttrium tetramethylhexadione ($Y(C_{11}H_{19}O_2)_3$ or $Y(tmhd)_3$) precursor [Strem Chemicals, CAS 15632-39-0] into a fine powder using a mortar and pestle, and then coating $\frac{1}{4}$" diameter stainless steel ball bearings with the precursor to increase the surface area for sublimation of the precursor. A drybox is used for evaporator packing due to the moisture sensitivity of the precursor described later. When an evaporator is reattached to the reactor system, it is pumped down to a base pressure of 0.6 torr. If an evaporator is not pumped down, a large pressure buildup inside the evaporator will occur as it is heated to the sublimation temperatures. This pressure buildup, upon opening the evaporator, could cause some of the precursor powder to blow out of the evaporator and into the reactor system, causing loss of precursor.

Except for the MFCs and the flow manifold, everything else in the upstream section is housed inside a hot air convection oven [Grieve NB-350] to keep the system at a constant temperature and to prevent precursor condensation in the tubing. The oven temperature is set manually, but is monitored using a thermocouple.

The concentration of the sublimated metalorganic precursor in the evaporator were calculated using the ideal gas law and Raoult's law and shown in Table 1. The vapor pressure of yttria precursor is reported in [49]. This is only a rough calculation of how much could be in the gas phase of the evaporator at one time. Chou et al. analyzed the evaporation of tmhd precursors and listed some factors that affected precursor evaporation rates [33]. The evaporation rates decreased over time due to a decrease in the effective surface area and an increase in the diffusion distance in the

Table 1: $Y(tmhd)_3$ precursor physical properties and concentration determination.

property	units	value
molecular weight	$\frac{g}{mol}$	638.71
melting point	^oC	170–173
sublimation temperature	K	413
gas constant R	$\frac{L\text{mm Hg}}{\text{gmol}K}$	62.36
vapor pressure	torr	0.062
total pressure in reactor	torr	1.5
calculated mole fraction of precursor	–	0.0413
calculated concentration of precursor in evaporator	$\frac{\mu mol}{L}$	2.4

region above the sample in the evaporator. For these reasons ball bearings are coated with the precursor to increase surface area as much as possible and the carrier gases are directed into the evaporator to minimize the diffusion length for the sublimated molecules.

To monitor the molar flowrate of precursor to the reactor chamber a UV cell connected to a spectrometer [Ocean Optics Model S2000] by fiber optic cable is used. Two channels of absorbance are used to monitor the precursor flow. Slave 1 is used to monitor the absorbance in the $Y(tmhd)_3$ UV cell and Slave 2 is used to monitor the stability of the UV lamp and is connected directly to the light source. $Y(tmhd)_3$ has a maximum absorbance at $\lambda_{max} = 286$ nm. This wavelength is used to calculate the concentration using the Beer-Lambert relation, $c = \frac{A}{\epsilon b}$, where c is the concentration in $\frac{mol}{L}$, A is the absorbance, ϵ is the molar absorptivity of $Y(tmhd)_3$ in $\frac{L}{mol\ cm}$ and has a value of 34673.7 [38, 106], and b is the length of the UV cell which is 10 cm. With a known concentration and molar flowrate of argon, n_{Ar}, one can calculate the molar flowrate of $Y(tmhd)_3$, n_Y. Using the ideal gas law, $c = \frac{n_Y}{V} = \frac{n_Y}{(n_Y+n_{Ar})\frac{RT}{P}}$ with R=62.3638 $\frac{L\times\text{mm Hg}}{mol\times K}$. Rearranging this equation yields

$$n_Y = \frac{1}{\frac{P}{RTc} - 1} n_{Ar} \tag{17}$$

Using Equation (17) one can monitor the precursor flow, to determine more accurately how much precursor has been used, and to determine when the evaporator needs

Table 2: Parameters for PID controller for precursor mass flow

K_P	0.00203
K_I	0.001
K_D	0.00
limit	$\pm$ 0.5 [sccm]

repacking. Monitoring precursor delivery is one of the problems in MOCVD that leads to repeatability issues [38, 106]. Using Equation (17) a PI controller for precursor delivery was designed. The equation used in the Labview software for converting absorbance to molar flow rate is

$$n_Y = \frac{A}{\frac{P\Gamma}{T} - A} n_{Ar} \Lambda \tag{18}$$

where $\Gamma = \frac{\epsilon \times b}{R} = 5560 \frac{K}{\text{mm Hg}}$ and $\Lambda = 7.4 \times 10^{-7} \frac{mol}{s \times \text{sccm}} \times 10^6 \frac{\mu mol}{mol} \times 60 \frac{s}{min} = 44.4 \frac{\mu mol}{min}$. Pressure, temperature, and absorbance are collected by the computer and used to calculate the molar flowrate every second. The manipulated variable is the argon gas flow and the parameters for the controller are shown in Table 2. Due to the noisy nature of the UV system, no derivative term was used, and additionally the controller was limited to varying the argon flow by 0.5 sccm in either direction during each time step.

To ensure that the amount of precursor coming out of the evaporator was constant from one evaporator packing to the next, the evaporators were tested after being freshly packed. The same pressure and temperatures were used, as well as similar flowrates through the evaporator and the chemical came from the same source bottle. The initial results are shown in Figure 6, and one can see the variability in the molar flowrate of precursor coming out of the evaporators.

A drybox was then used to ensure that the moisture in the surroundings and other uncontrollable environmental factors were constant from one packing to the next. The drybox is filled with nitrogen and the level of water vapor in the box is carefully tracked and was kept less than 0.1 ppm. Because of the strict control on water vapor, anything put into the drybox chamber needs to be baked in an oven at 130^{o}C for 3 hours. After taking the components for the evaporator out of the oven, they are immediately taken to the drybox and put in the antechamber and left under vacuum for at least one hour. Anything plastic which could not be baked in an oven was to be left in the antechamber under vacuum overnight before going into the drybox, to ensure that any water vapor be driven off. After packing the evaporators in the drybox environment the study was redone and the results are shown in Figure 7, where the molar flowrates are much more repeatable from one evaporator packing to the next. There is still some variability, especially at higher flowrates, but these variations are much less dramatic than in Figure 6 and can be compensated for by changing the flowrates or the oven temperature.

While it is important to know how much precursor is coming into the reactor chamber via the UV cells, not all of the precursor coming in will be incorporated into the deposited film. Precursor molecules can react prematurely and deposit on the walls of the reactor. The precursor may not come in contact with the substrate and instead is sucked into the liquid nitrogen trap and the pump. However, knowing that precursor is coming into the chamber is useful for monitoring the level of precursor in the evaporator, and is important when one wants to calculate how much of the precursor is being incorporated into the film to find the reactor efficiency.

Once the precursor passes through the UV cell, it is combined with another stream of argon and enters the flow manifold. The flow manifold is designed to precisely control the time at which the precursor goes into the reactor chamber. The manifold is made of four bellows-sealed valves [Swagelok] two of which are normally closed and two are normally open. The opening and closing of these valves is controlled using four solenoid valves [Swagelok] which are controlled by the computer software designed for the reactor [149]. The manifold has two inlets and two outlets. With no pneumatic pressure to the solenoid valves, the dummy argon flow goes to the reactor chamber while the stream containing the precursor bypasses the reaction chamber and goes directly to the outlet pump. When the pneumatic valves are on, the two streams switch. The flow manifold is outside of the oven, but the tubing is heated using heating tape [Omega]. The dummy argon flow and the flow manifold are used to continually flow gas through the chamber to minimize the variations in the gas flow inside the reactor. In practice, the flow switch is far from ideal and often leads to flow variations visible on the UV data.

The tubing for this reactor is placed within the oven whenever possible to ensure temperature uniformity and to prevent condensation of precursor in the lines. To ensure that the gas reaches thermal steady state before reaching the precursors, the length of tube needed to heat the gas to the appropriate temperature was calculated [131], with an extra 20% added to ensure gas temperature uniformity and to ensure there is no clogging in the tubes.

2.2 Deposition Chamber

The deposition chamber consists of two standard stainless steel ultra high vacuum reducing crosses [MDC Vacuum Products] which are also housed inside the oven to prevent precursor condensation. The main flange and the reducer flange are 4-$\frac{1}{2}$" OD and 2-$\frac{3}{4}$" respectively. The two flanges are rotated 90° from each other as shown in Figure 8. The bottom reducing cross has one reducing flange that has an outlet to the pump while the other reducing cross has a manual valve attached which is used to release pressure in case of system overpressuring. The top flange of the lower cross is attached to the upper cross, while the bottom flange is where the substrate heater and holder are inserted into the system. The top cross has one reducing flange, which is the inlet flow from the flow manifold. The other reducer flange has two tubes connected to it. One is for the oxygen and the other is used as a purge flow. Oxygen is needed for the precursor compound to decompose properly [74, 93] and is also a component of the film deposited. The flow of oxygen and the flow of precursor were separated to hinder any reaction of the precursor in the gas phase before reaching the deposition chamber. The top of the upper cross is attached to a viewport used with the emissivity correcting pyrometer (ECP) [SVT Associates] used to monitor *in situ* surface temperature and growth rate. The deposition chamber is also housed inside the oven to ensure temperature uniformity.

The type of deposition chamber for this study was chosen to be cold wall vertical flow reactor. This type is commonly used in rapid thermal processing and is best for the *in situ* sensing mentioned later. Cold wall reactors are characterized by heating the substrate directly while hot wall reactors heat the entire deposition chamber. Cold wall reactors can be advantageous because the precursor is more likely to deposit on the substrate and not on the walls because it is at a higher temperature than the walls of the reactor, which are heated in hot wall reactors.

To understand the flow patterns inside the deposition chamber the Reynolds number was calculated as well as the space time of the gases through the chamber. In addition modeling was performed with COMSOL. A Reynolds number for the gas flow in the deposition chamber was calculated using the Reynolds equation:

$$Re = \frac{D \times \phi \times \nu}{\mu} \tag{19}$$

where: $\phi [=] \frac{kg}{m^3} = 0.00677$, $D [=] m =$diameter, $\mu [=] Pa \times s = 0.0002099$ (conditions at 1.01 bar and 0^oC), and the velocity is calculated using a maximum flow of 700 sccm and the diameter of the tubing. ν is in units of $\frac{m}{s}$. For the main reactor which has $D = .0508m$, $\nu = 1.5\frac{m}{s}$, the Reynolds number = 24.7. For the $\frac{1}{2}$" tubing leading to reactor, $D = 0.0127$, $\nu = 24.1\frac{m}{s}$, the Reynolds number = 98.84. Thus the flow in the reactor is in the low laminar regime since the transition Re is 1200.

The volume of the reactor was calculated using an equation obtained from the

Table 3: Constants used in residence time calculations. Density and viscosity values for argon were used.

	value	units
sccm to moles conversion	7.4e-7	$\frac{moles}{sccm}$
molecular weight of argon	39.948	$\frac{g}{mole}$
ϕ (at reaction conditions)	0.1499	$\frac{m^3}{g}$
m^3 to cm^3	100^3	$\frac{cm^3}{m^3}$
calculated volume	2249	cm^3

ideal gas law. $PV = nRT \implies d(PV) = d(nRT) \implies d(PV) = FdtRT \implies$

$$\frac{dP}{dt} = \frac{FRT}{V} \tag{20}$$

where F is the flowrate from the mass flow controllers which was constant at 1000 sccm and where $\frac{dP}{dt}$ is a constant. The volume and the temperature of the reactor remain constant. The deposition chamber was pumped down, then the pump valve was closed leaving the chamber under vacuum. Argon gas was then flowed into the chamber at a constant rate, using a mass flow controller. Using the data from this experiment [149] along with Equation (20), the volume of the reactor was calculated to be 2249 cm^3. Then, using the conversion of mass flowrates to volumetric flowrates and other constants in Table 3, the space times at various flowrates were calculated $\tau = \frac{V}{v_o}$ where V is the reactor volume and v_o is the volumetric flowrate. The density used was calculated for reaction conditions using the ideal gas law, rather than STP conditions. The space times are shown in Table 4 and represent on average how long a particle will spend in the reaction chamber. If the space time is too short, the precursor particles may not have time to react with the surface. Alternatively, if the space time is too long, the precursor particle may decompose and deposit somewhere other than the substrate surface.

The space time is useful to approximating how long a molecule will be in the reactor, but that is only under ideal flow conditions. Irregularities in the flow within the

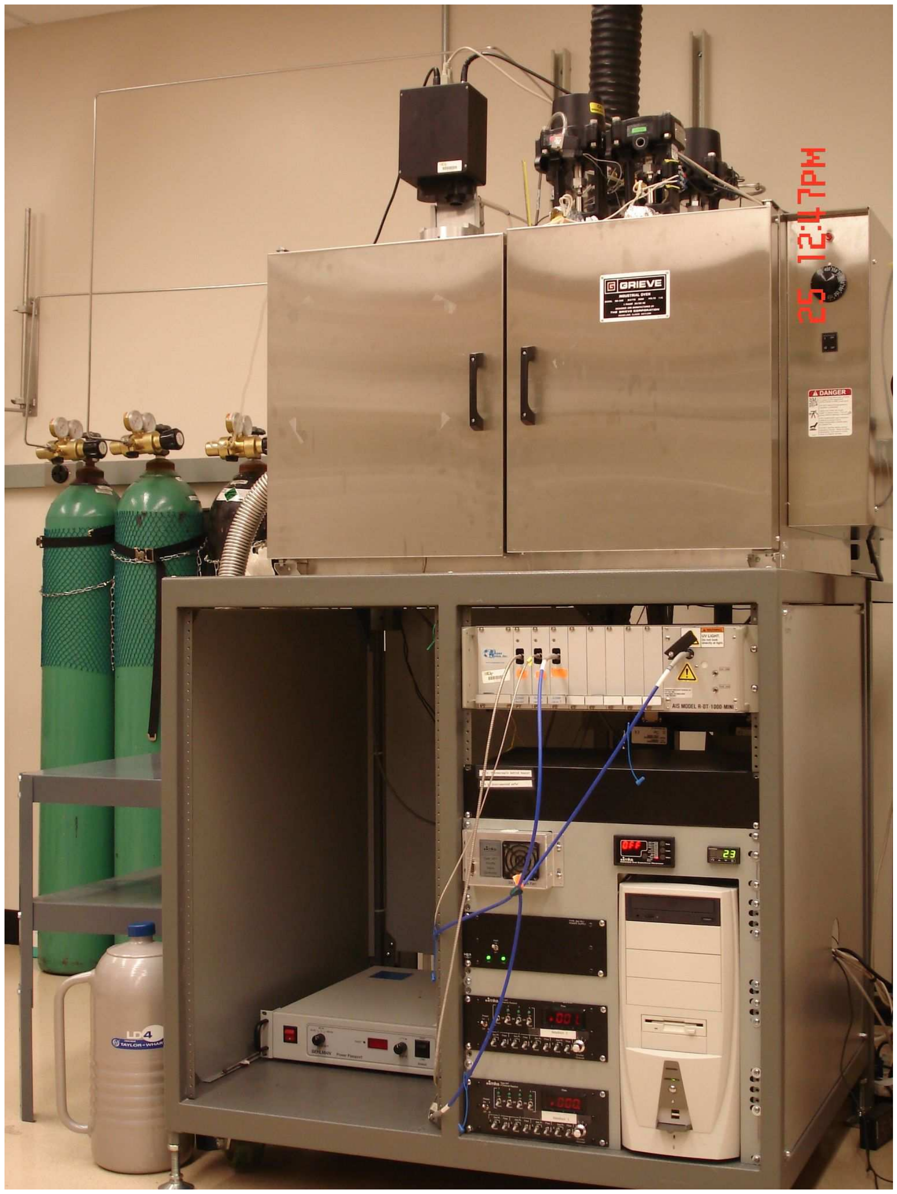


Figure 5: Picture of CVD system with oven doors closed.

Table 4: Space times at different flowrates through the reactor chamber.

Flowrates [sccm]	space times [s]
700	0.725
450	1.128
350	1.450
250	2.030
200	2.537

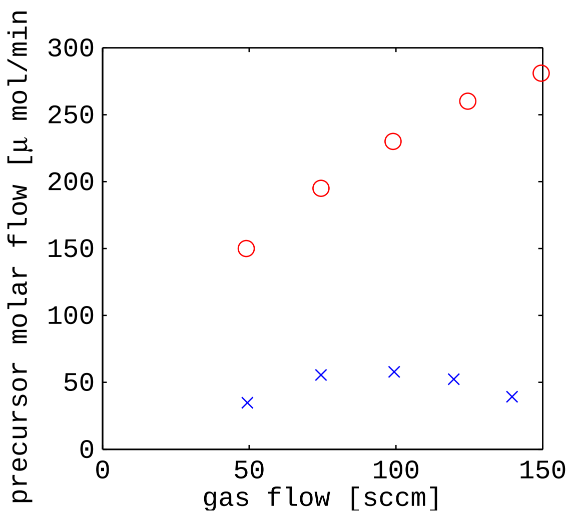


Figure 6: Results of initial study on precursor molar flowrate repeatability. The '×' markers are from the first evaporator packing, the '○' markers are from the second evaporator packing.

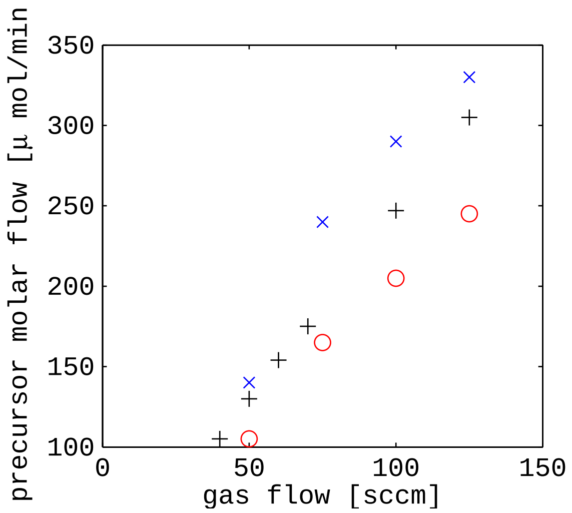


Figure 7: Results of precursor molar flowrate repeatability after using the drybox environment for packing. The '×' markers are from the first evaporator packing, the '○' markers are from the second evaporator packing, and the '+' are from the third evaporator packing.

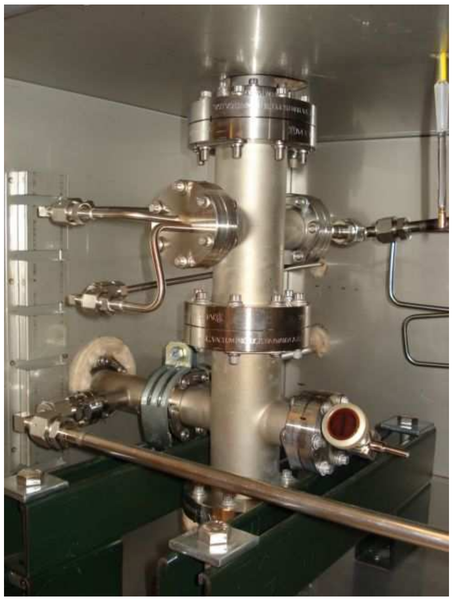


Figure 8: Picture of CVD deposition chamber.

reactor can be approximated using simulation programs such as COMSOL. COMSOL is a finite element calculation tool and is used to study the effects of flow non-uniformity in the reactor. A 2D model of the reactor was made for the calculations and is shown in Figure 9. Because this is a 2D representation of a 3D reactor, the heater assembly was not modelled but the substrate is floating at approximately the correct position as if it were sitting atop the heater assembly. Including the heater assembly in the 2D model would hinder any flow moving from left to right below the substrate due to the 2-D assumption. The pump outlet is the bottom right outlet of the reactor. The two upper flanges are where the inlets are located. The top left flange has two inlets, the window purge and the O_2 line. The window purge is modeled with an elbow pointing upward to see how running gases through this line would help the flow inside the reactor. The top right flange is the precursor inlet.

The goal of these simulations was to try to observe the effects of flow inside the reactor (i.e. how many stagnant regions were there, does the argon flow uniformly over the substrate, etc.). For this simulation the properties of argon gas were used,

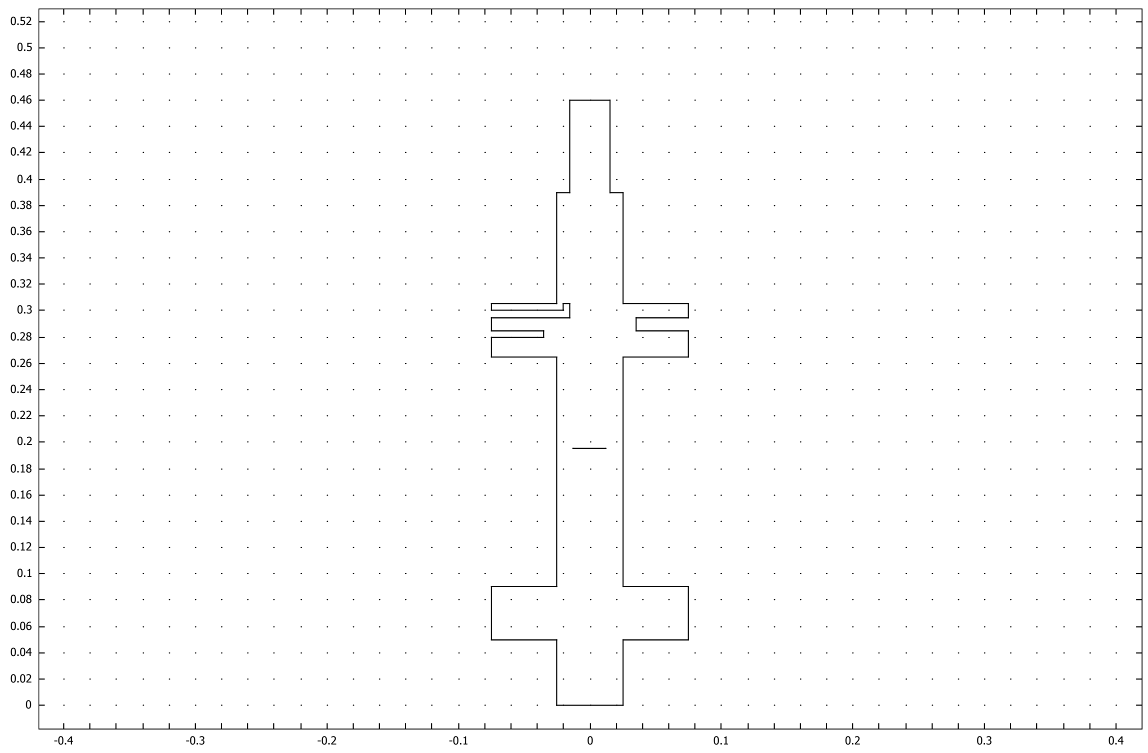


Figure 9: COMSOL drawing of 2D reactor–dimensions in meters

and the outlet was modeled to have a pressure of 2 torr or 266 Pa. The settings for each simulation are shown in Table 5, which lists the flowrates coming into the chamber from the three inlets: a $\frac{1}{4}$" window purge, a $\frac{1}{4}$" O_2 inlet, and a $\frac{1}{2}$" precursor inlet.

Model 1 was used to try to see what the flow would be like in the reactor if the O_2 and precursor lines were mixed before reaching the reactor. This actually results in one of the more uniform flows over the substrate, as seen in Figure 10(a). In this model, the O_2 was not turned on. The window purge line gets some flow out of the upper area of the chamber, but does not seem to affect the flow dramatically. An eddy forms beneath the precursor inlet and there is also an eddy in the bottom left flange of the reactor. These eddies can cause the precursor molecules to get trapped inside the reactor and not make it onto the substrate or to not make it out of the reactor during the expected space time. To ensure the reactor is clear of precursor

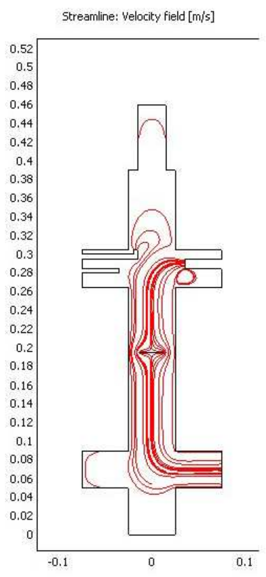
Table 5: COMSOL simulations

Model #	purge	O_2	Precursor	units
1	10	0	240	sccm
2	10	150	90	sccm
3	40	120	90	sccm
4	40	105	105	sccm
5	40	25	185	sccm
6	40	50	160	sccm
7	10	157	287	sccm

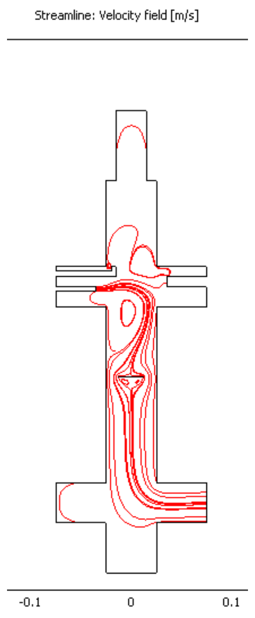
molecules after every experiment, it is pumped to a low pressure without any further precursor flow to allow the molecules to diffuse out.

Models 2–5 try various flow settings of oxygen to try to get rid of the stagnant region with all three inlets active. The results are shown in Figure 10(b) and show the formation of an additional recirculation zone above the substrate due to the oxygen flow. All of the results from Models 3–5 are very similar to Model 2. Model 6 is the result when both the O_2 and precursor inlets have very similar flowrates in $\frac{m}{s}$ and the resulting flow pattern is in Figure 10(c). Here the eddy under the O_2 is smaller but it appears that the mass flow from the precursor takes the gas over to the other side of the reactor before the gas flow is directed downward by the pump. This is indicated by the lack of flow lines from the precursor inlet. Model 7 is simulating what would happen at a total flowrate of 450 sccm with similar ratio of gas flows to the other experiments and is shown in Figure 10(d). The increased flow increases the size and number of recirculation zones in the reactors. For this reason the gas flows through the reactor were kept below 250 sccm.

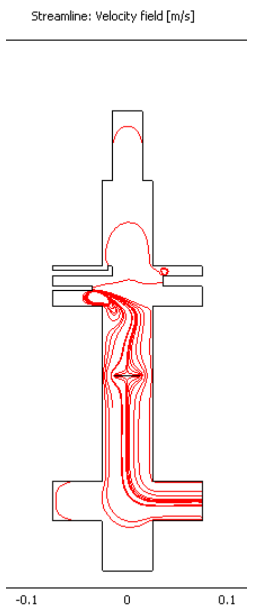
The design of this reactor causes many complicated flow patterns. First, most vertical flow reactors have the inlet at the top of the reactor and some use a showerhead to mix the reactant gases. In this reactor that design was not possible due to the line of sight needed for the reflectometer which is described later. Consequently, the inlets have to come in from the side of the reactor and rely on gravity and the pump



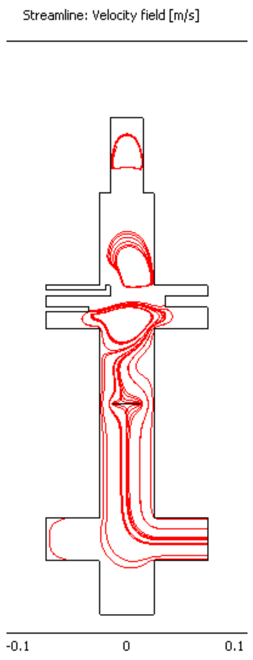
(a) Model 1



(b) Model 2



(c) Model 6



(d) Model 7

Figure 10: COMSOL results

to draw the particles toward the substrate and out of the reactor chamber. This leads to imperfect mixing of the O_2 and precursor flows. Second, the tubes used for the gases are just pointing into the chamber with no nozzle. The rapid expansion from the pipe to the reaction chamber causes abnormal flow patterns. The flow separates from the walls and recirculates causing a "Hamel-flow" vortex [118]. The eddies and vortices are nonuniformities that effectively increase the space time calculated previously. Third, the pump outlet for the reactor is to one side of the reactor causing uneven gas flow over the substrate. Again, because of the nature of the design, particularly the location of the heater assembly and the method of inserting and removing the substrate from the reaction chamber, this could not be avoided.

2.2.1 Heater Assembly

The heater assembly flange [Kurt J. Lesker] is shown in Figure 11 and is inserted into the lower flange of the deposition chamber and attached using an O-ring and a clamp. This flange contains the power feedthrough for the substrate heater [Heatwave Labs part: 101275-27] and a connection for a thermocouple attached to the back of the substrate heater. The substrate heater has a maximum temperature of 1200°C and the substrate is placed on top of the substrate heater and attached to the heater via clips. The substrate heater has wires which attach to the power feedthrough, so a ceramic coating [Cotronics Corp.] was used to protect these wires from the reactive gases in the chamber and to extend the life of the heater. The substrate heater is powered by an AC power supply [Behlman Electronics Model P1350] and the voltage output is controlled via the Labview software designed for our reactor system [149]. The heater voltage is increased gradually to protect the filament from burning out. A current sensor [Veris Industries] was added to monitor the current flow to the heater.

There were slight differences noticed in the voltage applied to the heater to reach a certain surface temperature. The power supply is rated to 1350 W of power, which

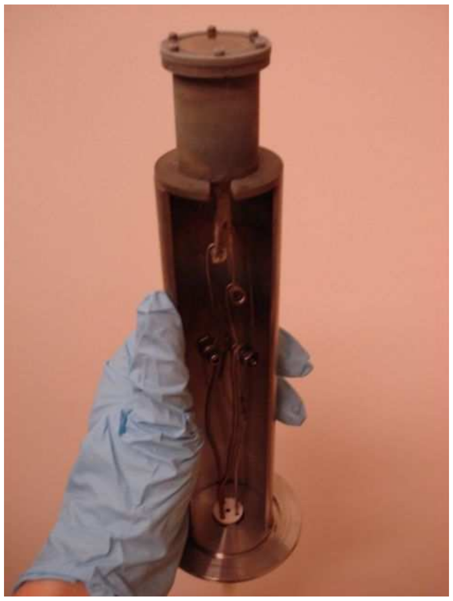


Figure 11: Picture of CVD heating assembly.

far exceeds the maximum power of the heater (242 W according to Heatwave Labs documentation). The total power ($W = V \times A$ varied slightly from one experiment to the next, suggesting the contact resistance between the substrate and the heater changed from one run to the next. The substrate clips used to affix the substrate to the heater may be applying different force on the substrate from one experiment to the next, causing the contact area to change. The substrate has also been known to deform at high temperatures [61] which could also cause variation.

The substrates used in this study were circular with 1" diameter, (100) orientation, p-type doped, prime grade silicon wafers with a flat on one edge [Wafer World Inc.]. The flat edge was very helpful in determining the substrate orientation on the heater. This was important especially when determining the direction of film thickness non-uniformities. The substrates were RCA cleaned and blown dry with N_2 before being clipped to the heater assembly. The full RCA clean can be found in Appendix A. Extreme care was taken during the RCA clean due to the use of HF in the etch step. When handling the HF bottle thick neoprene gloves, a labcoat, and lab glasses are worn and all mixing of chemicals is performed in the fume hood. In addition, the fume hood door is closed to protect the face during mixing. To perform the RCA clean and to minimize contact with the HF solution during the procedure, a wafer dipper [Entegris D11-0215] was used.

The surface temperature of the substrate is usually reported in literature as the temperature of the thermocouple attached to the back of the heater or the method of finding the surface temperature is not mentioned at all [50, 80]. Due to heater-substrate heat transfer, the actual substrate surface temperature is much lower than the temperature reported by this thermocouple. This can lead to issues when interpreting the data, especially if one is trying to gain an understanding of the underlying physics of the reaction. First, the reaction activation energies will be incorrect. Second, reporting the temperature from the back of the heater leads to repeatability problems. One CVD system will not necessarily have the same thermocouple placement as another system. Even if one tries to place the thermocouple in the same place, the thermocouple could become dislodged during handling leading to drift in the temperature values which would further complicate temperature data interpretation. While this thermocouple does indeed provide useful information especially if other equipment fails, it is not an ideal method for reporting the surface temperature of the system. Some researchers use a thermocouple placed on the substrate under simulated reaction conditions [142] to gain an idea of the surface conditions during deposition, but this can also be inaccurate due to heater drift from one run to the next. With the use of the emissivity correcting pyrometer on our equipment, we have much more accurate surface temperatures, so the small changes in resistance of the heater can be corrected by adding more voltage. In this way the *in situ* sensor is helping us to keep our reaction conditions more repeatable and insensitive to heater drift as well as for final film thickness.

To make the surface temperature data more repeatable, an emissivity-correcting pyrometer (ECP) [SVT Associates Model: In-Situ 4000] is used to monitor the surface temperature as in [21, 144]. ECP works by measuring thermal radiation intensity at 950 and 850 nm and normal reflected light intensity at 950 and 470 nm. The theory of ECP will be briefly described here but for a more in-depth description of the theory and our implementation the reader is directed elsewhere [149]. A real surface's thermal radiation can be described using blackbody radiation multiplied by the emissivity of the surface. As a surface adds layers (i.e. during deposition) the emissivity of the surface changes. The emissivity, ϵ, is measured independently using the reflectance, R, at the same wavelength, $\epsilon = 1 - R$. The ECP is mounted on a quartz viewport [MDC Vacuum Products] that is extended outside the oven using a straight tube which is attached to the upper flange of the deposition chamber. To minimize the area the ECP views and to simplify sensor data interpretation a cover is used on the ECP lens to reduce the size of the lens to $\frac{1}{4}$" instead of 1".

2.3 Downstream

The downstream portion of the system is located outside of the oven and consists of a pressure transducer, a liquid nitrogen trap, a pump, and the tubing to connect the three. The pressure transducer measures from 1 to 760 torr and passes this information to the pressure controller which adjusts the throttle valve to control the pressure at a certain setpoint [MKS Instruments] via the LabVIEW program for the equipment. Since this pressure instrument has a wide range, another pressure sensor [MKS PDR2000] was needed at lower pressure ranges. This sensor works from 0.001 to 10 torr and is very useful in leak detection and ensuring an accurate pressure reading at low pressures. Since this pressure sensor is more precise at low temperatures, the reading from this sensor is recorded in the experimental settings for the experiment. The liquid nitrogen trap is used to condense the vapors from the deposition chamber before the vapors reach the pump. The pump [Alcatel 2063 SD] has a pumping speed of 40-50 cfm to provide pressures as low as 1 torr at flowrates up to 1000 sccm.

The procedure for running the CVD system has gone through numerous modifications as equipment has been added to the reactor and as understanding of sources of variability matured. Above all, the reactor was analyzed before changing any procedures to make sure no unnecessary safety risks were being taken. The most up to date procedure for the CVD system can be found in Appendix B.

2.4 Thin Film Analysis

Once the films are deposited on the substrate, they need to be characterized *ex situ* to find the microstructural properties of the resulting film. The main techniques used were spectroscopic ellipsometry, x-ray diffraction (XRD), and atomic force microscopy (AFM).

Spectroscopic ellipsometry [J.A. Woollam Co., Inc. M-2000VI] was performed at three angles (65, 70, and 75^o) to find the thickness of the thin film. Ellipsommetry uses a wavelength of light, shines it at an angle across a substrate, and the light coming off the substrate hits a detector. The wavelength of light changes when it goes through another medium (i.e. the thin film) and one can approximate the thickness of the film using this change in wavelength. The software for the program enables the user to build a model of the thin film, and the software fits the model to the light measured from the substrate using user-specified parameters. A typical model is composed of a silicon wafer layer, a Cauchy layer (referring to the Cauchy model which is the mathematical form used to fit the refractive index of the thin film), and a roughness layer. The adjustable parameters were the thickness of the thin film and the A_n coefficient of the Cauchy model. The roughness was set to 20 nm and models roughness by having a 50% void in the roughness layer. A typical fit with the model to experimental data is shown in Figure 12.

The atomic force microscope is used [Agilent PicoSPM II microscope] to determine

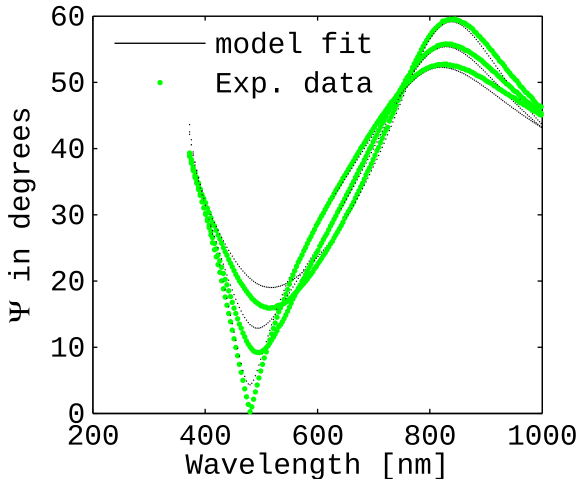
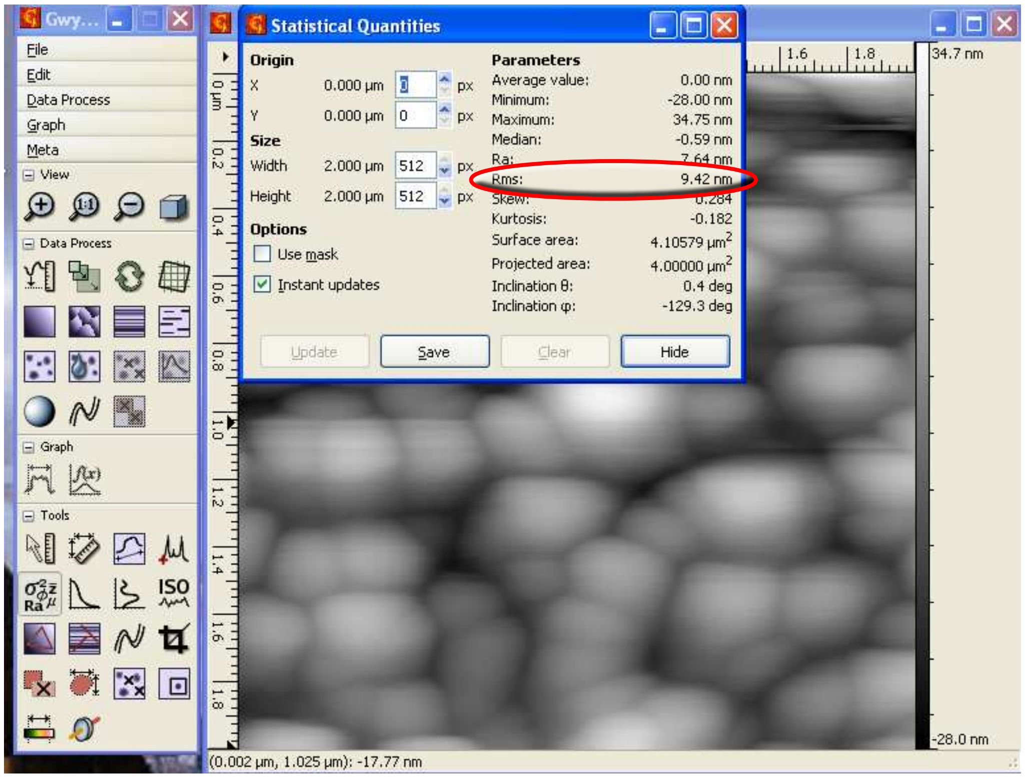


Figure 12: Typical model fit to ellipsometry raw data. The three lines are for the three angles used (65, 70, and 75^o), and the solid lines are for the model fit to the data.

the roughness and grain size of the thin films. The microscope is placed on an air table to reduce noise from vibrations. Non-contact mode AFM tips [Nanoandmore ARROW-NCR-10] are used in air in the analysis. A contact AFM imaging tip moves along the surface and bends when there are features on the surface that it comes in contact with. A non-contact imaging tip is oscillated at the resonance frequency above the surface, when the tip is near a surface feature the oscillation of the tip changes. By tracking these changes an image is constructed using the AFM imaging software for the microscope. For a more detailed explanation of the operation of an AFM see [54]. For this analysis, only non-contact imaging was used for all the characterizations. The images are 2 μm × 2 μm, and were taken with a scan speed of 1 line/second. At least three images were taken from different positions on the substrate to ensure that an accurate sampling of the surface was obtained. Images are processed with the Gwyddion software [gwyddion.net] to correct for tip artefacts by removing the polynomial background using a polynomial of the 3rd degree. The RMS roughness was then obtained using the statistical properties module (see Figure 13), and the results from different scans from the same surface were averaged. An



Statistical Quantities

Origin

X	0.000 µm	0	px
Y	0.000 µm	0	px

Size

Width	2.000 µm	512	px
Height	2.000 µm	512	px

Options

[ ] Use mask
[✓] Instant updates

Parameters

Average value:	0.00 nm
Minimum:	-28.00 nm
Maximum:	34.75 nm
Median:	-0.59 nm
Ra:	7.64 nm
Rms:	9.42 nm
Skew:	0.284
Kurtosis:	-0.182
Surface area:	4.10579 µm^2
Projected area:	4.00000 µm^2
Inclination θ:	0.4 deg
Inclination φ:	-129.3 deg

(0.002 µm, 1.025 µm): -17.77 nm

Figure 13: A screenshot of Gwyddion, the AFM image processing software used in this study. The red circle depicts the surface roughness obtained using the statistical quantities tool after image smoothing.

image from the AFM on one of the sample films is shown in Figure 14.

XRD [Panalytical] was performed on the thin films to determine the microstructure of the thin films. An XRD scan on a silicon substrate is shown in Figure 15 and the 100 peak is visible at $2\theta = 20$. A Scan of a Y_2O_3 film is shown in Figure 16. The purpose of doing the XRD scan was to determine what phase the material was in and to optimize this microstructure as well. Unfortunately, identification of XRD scans is troublesome at best. There are many different microstructural configurations for ceramic materials and it is very difficult to positively identify any of the peaks much less the microstructure as a whole. This is caused by many reasons including a shift in the XRD scan either due to user error or from some artifact of the machine. After many weeks of training, attempted identification, and help from the available experts,

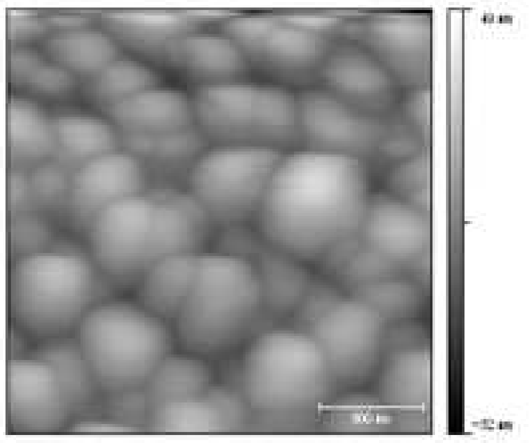
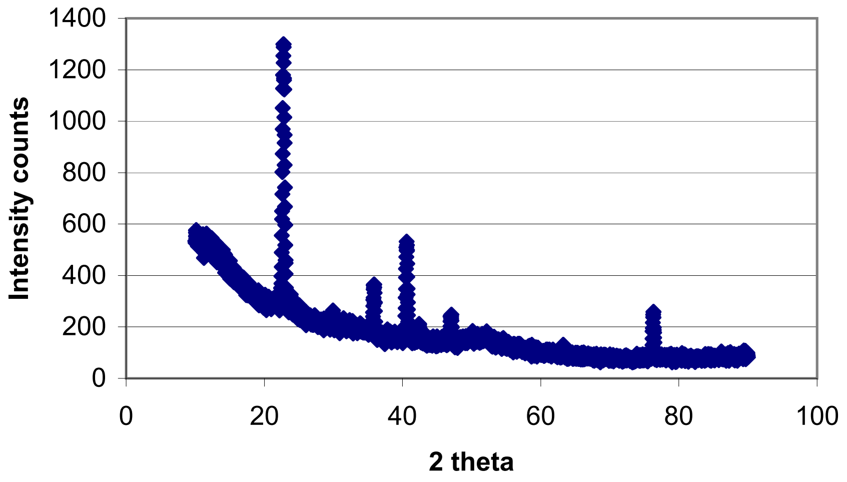


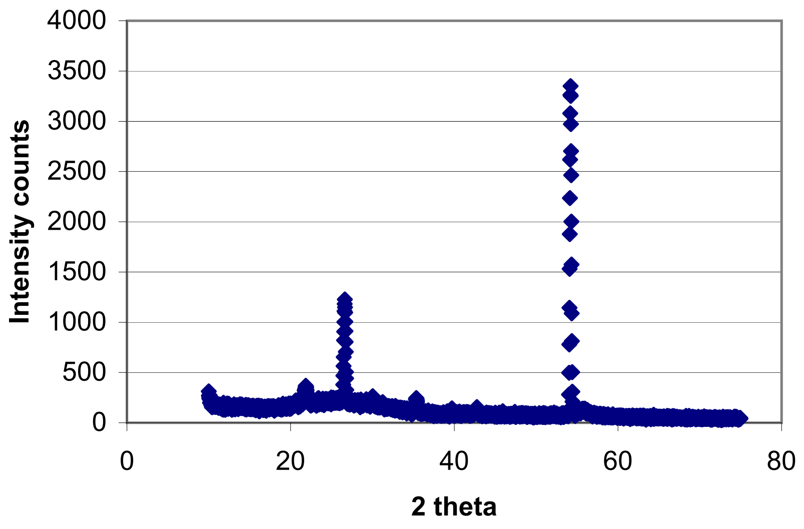
Figure 14: A 2 μm by 2 μm atomic force microscope image of a thin film grown in the reactor.

Figure 15: XRD scan of a silicon substrate



this analysis technique is abandoned and left for an expert to pursue later. The focus of the microstructural models developed was not on crystallographic structure but on roughness and grain size. Therefore XRD data was not used in developing the models in this thesis.

Figure 16: XRD scan of a yttrium oxide thin film.



CHAPTER III

METHODOLOGY FOR EXPERIMENTAL DESIGN

In this chapter, we will speak briefly about the different areas of experimental design. We will then develop the experimental design used in this thesis. This design incorporates aspects of many different experimental design techniques for the purpose of quickly finding the optimal point of a process and to develop a useful process model.

3.1 Past Work

Experimental design is an effective way to conduct a comparison between treatments in terms of a recorded response [92]. Different treatments are made using controlled variables or "factors" which can be controlled during the experiment. Different values or "levels" for each factor are chosen and different experiments are run for each level of a factor. The response variables are measurements or observations made after changing the factors. The effect a factor has on the observation is termed a "main effect". Factorial and fractional factorial experiments are a commonly used form of experimental design used in industry and research. It is easy to implement in a production setting and the conclusions one can draw from the experiments are statistically sound.

Factorial experimental design is an enhancement of the one factor at a time experimentation that is sometimes used in research studies. One factor at a time experiments determine the effect of a factor on a response by changing that factor alone. However, it is often important not only to determine if factors have an influence on the response, but also determine if there is significant interaction between the different factors [140], and factorial experiments address this issue. A factorial experiment is where all combinations of the factors are explored and more than one factor are varied

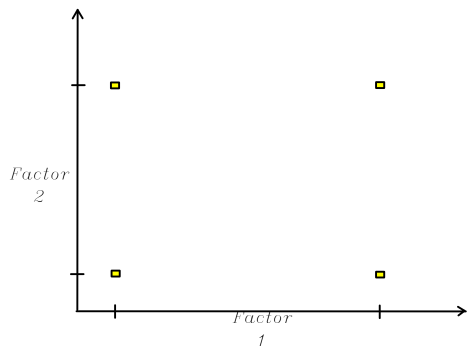


Figure 17: Design of experiments example with two factors. The experiments are shown as boxes placed at the high and low settings of the factors to elucidate the interaction between factors.

at a time to analyze the interaction between factors. An example of a 2^2 factorial (two factors with two levels each) experimental design is shown in Figure 17. To gain the necessary interaction information from the one factor at a time approach, more experiments would be needed than the factorial experimental design. A fractional factorial design is a carefully chosen subset of the factorial experiments used when the number of factors and levels is large compared with the resources available (time available to run experiments or limited amount of reagent for reaction) to run the experiments.

Suppose you have a process and there are many factors affecting your process output such as temperature and flow rate. The process also has limits on each of these factors such as a maximum and minimum operating temperature and maximum and minimum flowrates to meet production requirements. The high and low level of a factor are usually put on a coded scale from -1 to +1 which make the analysis of the effects more straightforward. Using one factor at a time experiments, each factor is varied from its current setpoint independently of the other factors and four experiments would be run, but the interaction between factors would not be analyzed. Using a factorial experimental design, one would still run four experiments, but would not only have the information on the effect of the individual factors on the process output, but also information on the interaction between the two factors that would be missed using one factor at a time methodology. Plus, one would have data over a larger range of experimental conditions making factorial designs more efficient and ultimately more effective than one factor at a time experiments [84].

Another important development in experimental design came from Taguchi which was introduced in the United States in the 1980s [84]. The Taguchi method is an experimental design aimed at making a process more robust to noise factors. The main idea of the work was to identify the factors that cause variability in the process outputs and to design experiments to minimize this variability [99]. These experiments are sometimes called crossed array designs because they design the experiments using process factors in an inner array and the noise factors in an outer array. A noise factor may or may not be controlled in the experiment, but are not controlled in the day-to-day operation of the process. By varying the controllable factors, the operating point for the process is found which minimizes the variability caused by the noise factors. The actual experimental designs developed by Taguchi has received much scrutiny in statistics and engineering fields due to sometimes inefficient and ineffective designs [84]. However, the concepts developed to solve the robust parameter design problem are very important and is still an interesting and active area of research [3]. Among his contributions was that his work influenced engineers and statisticians to rethink statistical methods in terms of sensitivity to environmental variables as well as prompting the use of product and process variability as an important part of process performance criterion [140].

Robustness has also been a very active area of modeling research. In chemical engineering the research has been focused on robustness of process models for use in process control [1, 4, 6, 17, 75]. Other work in the engineering field has focused on designs which are robust to uncertainty [10, 69]. This uncertainty is quantified using an imprecise probability which is constructed using information economics. "Information economics is the study of choice in information collection and management when resources to expend on information are scarce" [69]. Experimental design is a form of information economics as one is trying to construct experiments using the least amount of resources possible, but this work also includes the management decision of when the cost of more experiments outwieghs the information gain from new experiments and experiments are stopped.

Response surface methodology (RSM) was developed to deal with the ineffiecencies identified in the Taguchi method [84]. RSM is a sequential experimental procedure that uses factorial, fractional factorial, or other experimental designs to identify the optimum of a process [88]. An example of this methodology is shown in Figure 18 where a factorial experiment plus a center point have been run and are shown as boxes. This first set of experiments are designed to help find the optimal point. Once the optimal point is found, the next set of experiments (shown as circles) are designed in the vicinity of the optimal point to acquire more data around the optimal point. In this example, the optimal is within the initial experimental region, but this is not always the case in response surface methodology. This procedure of experimentation, optimization, then more experiments is repeated until the goals of the experiments are achieved. The main drawback of this approach is its empirical nature. The model is found using relations in the experimental data and do not relate theory to explain the experimental results. RSM works well to optimize an objective function, but what if the objective function changes due to changing costs or changing specifications on the product? Changing the process settings may cause some of the data acquired to be useless in finding the new optimal point for the process whereas a mechanistically-based model would allow for extrapolation to other experimental regions.

RSM is also a good example of sequential experimental design. Even if a set of

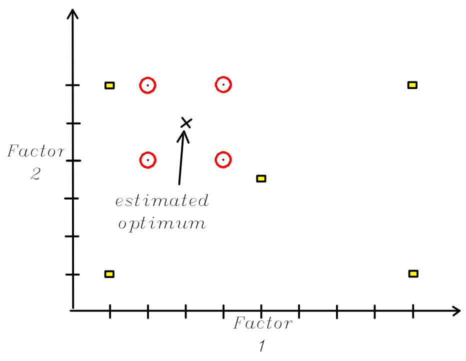


Figure 18: Example of response surface methodology. The first set of experiments are shown as boxes, the second set of experiments are shown as circles. The second set of experiments are planned in the vicinity of the estimated optimal point of the process.

experiments is impeccably designed, experimental results likely lead to more questions than answers about the process requiring more experiments if the resources are available to do so. One or all of the experimental designs mentioned here can be used sequentially. A good example of this is in [91, 90], where the researchers perform a factorial design, then design the next set of experiments using the results from the first set of experiments.

Methods for developing mechanistically-based hybrid models focus more on parameter estimation. The base model is derived from first principles or from prior knowledge of the system, but the knowledge is sometimes incomplete and parameters need to be estimated from experimental data. The focus for experimental designs for these types of models is accurately estimating the unknown parameters of the model [53, 79]. These designs typically employ D-optimal experimental design, which minimizes the variance of the model parameter estimates.

$$D_{opt} = \min|(X'X)^{-1}| \tag{21}$$

D-optimal has several variations itself including D$_s$ optimal design [124] and DS optimal design [70]. Other optimal design theories are A-optimal

$$A_{opt} = \min tr((X'X)^{-1}) \tag{22}$$

which minimizes the variance of the model parameter estimates. E-optimal experimental design minimizes the maximum eigenvalue in $(X'X)^{-1}$. Other experimental designs make use of the prediction variance from Equation (11) such as G-optimal which minimizes the maximum prediction variance over the design space

$$G_{opt} = min(max(\mathbf{a^{(i)}}(\mathbf{X_i'X_i})^{-1}\mathbf{a^{(i)}}'\sigma^2)) \tag{23}$$

V-optimal is similar to G-optimal but minimizes the maximum prediction variance over only a set of points in the design space. These optimal experiments improve the certainty of the fitted parameters but this is only useful if the model developed is the correct one. One can design experiments to get better estimates of the model parameters, but if the model is incorrect, having precise model parameters does not help.

Another experimental design method of interest in current research is Bayesian experimental design. Bayesian solutions change in a sensible way when the prior probability distribution and the utility function change [28]. This has been implemented in a Bayesian D-optimal design [27, 41] where the utility function for the experimental design is constructed to give the best estimation of the model parameters. Bayesian experimental design has also been used alone as a method of designing experiments [37, 62, 87]. A Bayesian calibration method for computer models has been introduced that includes model inadequacy in the probability calculation [63]. For systems where little prior information is available, optimal Bayesian designs correspond to non-Bayesian experimental designs. When non-informative prior probability distributions are used, the Bayesian approach does not offer an advantage. Since little was known about our system of interest and we wished to rely on the experimental data, informative prior probabilities were not used and Bayesian statistics were not used in the experimental design methodology development.

In addition to experimental designs which assist in constructing a model and improving a model, there are also experimental designs to assist in discriminating

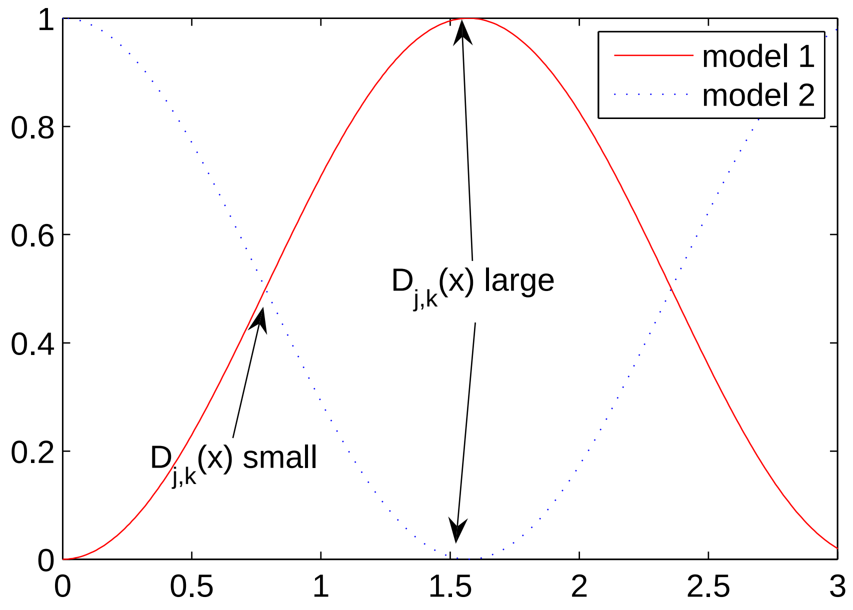


Figure 19: The discrimination function is large where two models' predictions disagree the most.

between multiple possible models. Box et al. was one of the first to tackle this problem and used a probability associated with each model to find the best model. However this was developed to distinguish between models using existing data and not specifically for designing new experiments to distinguish between possible models [18]. The concept of model discrimination was extended to experimental design by Atkinson et al. Initial work by Atkinson in this area focused on designing experiments to detect poor regression models [9], but was later extended to designing experiments to discriminate between different models [7, 8]. The idea was to design experiments where one would maximize the difference in predictions of the models. In Figure 19, one can see that the discrimination function designs experiments where two models' predictions disagree the most. By running experiments at this point, one can determine which model fits your process and experimental region best. An important assumption in [7, 8] was that one of the models is the correct model. However this does not help an experimenter with a new process which probably will not have the true model available. Can one still design experiments to see which one is best or to find if both the models are poor and a new model is needed? This is one of the key questions we hope to answer in designing the new methodology in this chapter.

3.2 Methodology

The focus here is to combine the best features of various methods for experimental design that are currently being used. By combining the methods for empirical and mechanistic models, one could efficiently find an optimal point, while also building mechanistic models that are useful in explaining the phenomena behind the process. Here optimal is defined as minimizing or maximizing some predefined continuous objective function, but other definitions of optimal are also possible, such as designing the process to perform within a certain interval. The insight gained from these experiments can be used for future work, whether it be to re-optimize the process around a different operating regime or to design a new process. At best, one or more of the models will have a good prediction over the entire region, such that it can be used for process design over the entire experimental range in question, although this will not necessarily be achieved.

The basic steps in our proposed methodology are illustrated by Figure 20. One begins with an initial probability of each model M_j, where $j = 1, 2, ..m$, and a hypothesis as to which experimental settings, x_r, $r = 1, 2, ..q$, should be varied to reach the design goal. The next step is to design an experiment or set of experiments to acquire some initial information about the system to obtain initial estimates of the unknown parameters θ_j in each model.

After each sequential experiment has been performed, the parameters for each model are reestimated using a parameter estimation technique. In this work, the parameters are estimated using a least squares minimization between the model and all of the data. Once the parameter estimate, $\hat{\theta}$, has been obtained for each model,

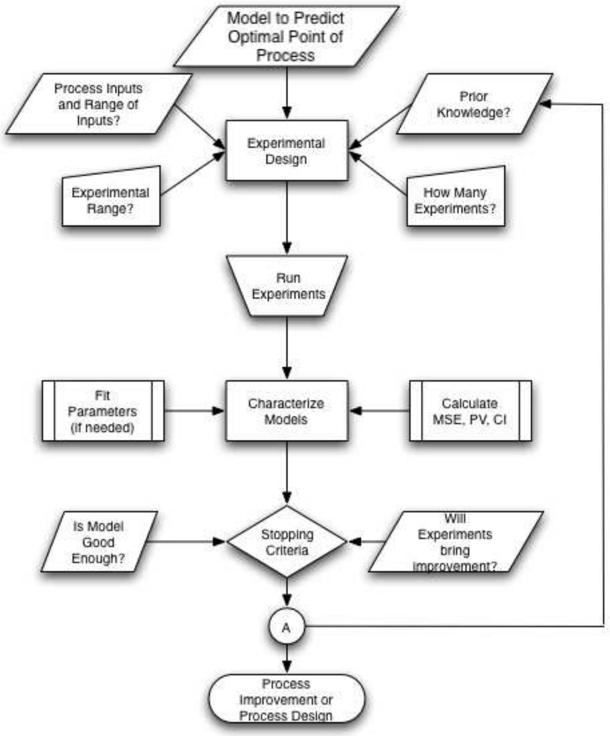

Model to Predict Optimal Point of Process
Process Inputs and Range of Inputs?
Prior Knowledge?
Experimental Design
Experimental Range?
How Many Experiments?
Run Experiments
Fit Parameters (if needed)
Characterize Models
Calculate MSE, PV, CI
Is Model Good Enough?
Stopping Criteria
Will Experiments bring improvement?
A
Process Improvement or Process Design

Figure 20: Proposed steps of experimental design algorithm.

the performance of each model can then be examined. The optimal point for each model is estimated as

$$\hat{x}_j = \min_x f(\hat{y}_j(x)) \tag{24}$$

where $f(x)$ is the objective function to be minimized. Note that $\hat{x}_j$ is the estimated optimal point from the *model* j and not necessarily $\bar{x}$, the true optimal point of the *process*. As a model becomes more accurate, $\hat{x}_j$ and $\bar{x}$ will ideally converge to the same value.

The probability of each model is calculated using Eqn. (7), and here we set the *a priori* probabilities $P(M_j)$ to be equal for all models. The prediction variance $\sigma_j^2(x)$ is used to calculate the confidence interval

$$CI_j(x) = \pm t_{\alpha/2,n-p_j}\sqrt{\sigma_j^2(x)} \tag{25}$$

where α is the level of confidence desired [84]. Once the performance of each model has been calculated, the stopping criteria are used to evaluate whether to continue the experiments or not. The stopping criteria used here are evaluated using the most probable model, j^*. The first criterion considers whether the change in MSE of model j^* is less than ϵ_{tol}, where ϵ_{tol} is a prespecified constant value. The second criterion checks whether the confidence interval at $\hat{x}_{j^*}$ is below the desired level, ϵ_{CI}. If the first stopping criterion is true, then additional experiments are unlikely to improve the design or the confidence interval on the model prediction. If either criterion is true, then the experiments are finished and intervention by the experimenter is needed to interpret the results. The experimenter must decide whether the most probable model is good enough, if this model needs modification, or if an entirely new model is needed. At any time during the experiments, it is possible to add one or more new models and continue to iterate through the methodology.

The basic steps of the methodology are

1. Define purpose of the model
2. Find which inputs to process affect the output (screening experiment)
3. Define objective function
4. Define stopping criterion
5. Choose experimental design
6. Run experiments
7. Fit parameters (if needed) and characterize models
8. Use stopping criterion to decide whether more experiments are needed
9. Add to possible models and/or change existing models using knowledge gained from previous experiments
10. Continue experiments

3.3 Initial Experimental Design Work

For the intial work, a method for discriminating among multiple possible models was used to optimize a process and design experiments sequentially. In [24], the next experiment is chosen based on Equation (26). The function $D_{j,k}(\mathbf{x})$ is maximized over the experimental inputs $\mathbf{x}$ to find the best experimental setting to discriminate between the models. This equation picks the best experimental point based on the difference of two models' predictions and weights it using the prediction variance of each model:

$$D_{j,k}(x) = (P_j P_k)\frac{[\hat{y}_j(x) - \hat{y}_k(x)]^2}{2\sigma^2(x) + \sigma_j^2(x) + \sigma_k^2(x)} \tag{26}$$

where $D_{j,k}(x)$ is the discrimination function between models j and k, $\hat{y}_j$ is the prediction of model j, and σ is the experimental variance. The prediction variance of model j at settings x as calculated by Box and Hill [18] is $\sigma_j^2(x)$. The prediction variance is a measure of the uncertainty of the model

$$\sigma_j{}^2(x) = \mathbf{a^{(j)}}(\mathbf{X_j'X_j})^{-1}\mathbf{a^{(j)}}'\hat{\sigma}_j^2 \tag{27}$$

The model variance $\hat{\sigma}_j^2$ is used since the experimental variance is generally not known *a priori*. It is calculated in Equation (27) is calculated similarly to MSE_j in Eqn. (8), except the summation is divided by $n - p_j$, where p_j is the number of estimated parameters in model j because p_j degrees of freedom are lost by estimating p_j parameters.

$$\hat{\sigma}_j^2 = \frac{\sum_{i=1}^{n}(y(x_i) - \hat{y}_j(x_i))^2}{n - p_j} \tag{28}$$

In the methodology attempted initially, the two distinct approaches used by Box and by Buzzi-Ferraris were combined using the framework of model discrimination. An objective function was proposed that accounts for system performance, based on the probabilities of the candidate models. Both empirical and mechanistic models were included. Initially, it was expected that the empirical model will be more likely if it has fewer parameters, but after many experiments, the mechanistic model may become more probable and therefore be more useful for process optimization.

The case study for this method is from thin film deposition (a microelectronics process). Chemical vapor deposition (CVD) is a commonly used process, where the properties of the thin film are determined by the microstructure which in turn is determined by the processing conditions. Currently, this process is not well understood and much time and resources in industry are spent finding the correct recipe for the deposition process [42]. A well-developed model could reduce optimization time and therefore increase productivity of microelectronics fabrication facilities. Using our own customized CVD reactor, we are building a model to predict microstructure from processing conditions of the reactor.

The goal of the initial case study is to compare the usefulness of a mechanistic and an empirical model in the early "nucleation" phase of the deposition process [43]. Both models have a number of parameters that must be estimated from measurements. Computer experiments are then run to estimate parameters for this nucleation model, as well as the empirical fitted model. Once the initial set of experiments has been run, a sequential experimental design approach is followed which uses an objective function that chooses the next set of experiments to distinguish between the models while also minimizing nucleation density.

The primary goal of the experiments is to compute the optimal settings of our process to obtain a desired microstructure. The "best" microstructure for one use may not necessarily be the best microstructure for another use. For example, yttria-stabilized-zirconia (YSZ) is used as a thermal barrier coating (TBC) for turbine blades. In this application, many grain boundaries are required to reduce the thermal conductivity of the coating. On the other hand, YSZ used for solid oxide fuel cells electrodes are needed that have large grains (i.e. few grain boundaries) as one wants a high conductivity in that application. For different applications, different microstructures are needed. While doing an optimization of the process parameters to pick the best settings for a particular application, it is important to include the design objective in the selection of experiments.

Building upon the concepts presented earlier, we propose a discrimination function of similar form as that described by Buzzi-Ferraris et al.:

$$\bar{D}_{j,k}(x) = (P_j * (f(\hat{y}_j(x))) + P_k * (f(\hat{y}_k(x)))) \frac{[\hat{y}_j(x) - \hat{y}_k(x)]^2}{2\sigma^2(x) + \sigma_j^2(x) + \sigma_k^2(x)} \tag{29}$$

where $f(\hat{y}_j(x))$ is the objective function for the process, as predicted by model j. The objective function contains the properties of the material one would like to optimize. This new discrimination function contains the original function in equation (26). We modify (26) by multiplying by an additional term, which takes into account the predicted *performance* of the system. When only two models are considered, this new portion is simply the expected value of the objective function, f.

This $\bar{D}$ selects new experiments to discriminate between candidate models, but also to concentrate the experiments near the optimal performance point. As a result, this sequential experimental design approach *simultaneously* builds models, validates them, and finds the optimal settings. As opposed to previous work in this area, we propose to update the "best fit" model parameters as more experimental data is obtained. For this, if a model probability is initially very low, we still want the discrimination portion of $\bar{D}$ to predict a next experiment. It may be that the model's probability will increase with more experimental data (as can be the case with mechanistic models). If a model is predicting good performance in the desired range, then that probability will be weighted in favor of the model. Depending on the application, the form of the objective function f will change.

Simulations were run using this discrimination design and are described in Chapter 4. Given more than two models, we observe that the discrimination function tends to choose the models which disagree most, usually the most probable model and the least probable model. Designing the next experiment using the worst model was seen as undesirable. Also, just because two models disagree at a point does not mean that the point will be useful in finding the optimal point of the process. The two models may disagree at a point far away from the optimal point of the process. For these reasons the work based on discrimination functions was halted and a new direction was chosen.

3.4 Revised Experimental Design Methodology

The methodology shown in Figure 20 remains unchanged after the departure from the discrimination function. The methodology was not the problem, but a different experimental design was needed to achieve our goals. Now, after the first set of experiments has been performed, we evaluate several types of experimental designs for the sequential portion. The three methods compared are D-optimal ($D_{opt} = \max_x |X_j{}' X_j|$)[84], a random design of experiments, and our newly proposed P-optimal design, which utilizes the prediction variance.

D-optimal and P-optimal are experimental designs with different objectives. D-optimal designs are frequently used for experimental design and minimize the generalized variance of the parameter estimates. In contrast, G-optimal designs attempt to minimize the maximum variance of the predicted values y over the entire experimental region [84], but are generally very computationally expensive and are less commonly used. Our P-optimal design is a modification of G-optimal. Rather than minimizing $\sigma_j^2(x)$ over the entire design region, P-optimal aims to minimize $\sigma_j^2(x)$ at a single point x^* (P=$\min_{x_e} \sigma_j^2(x^*)$), where x_e is the experimental point. Thus, P-optimal picks the experimental point which should most reduce $\sigma_{j^*}^2(x^*)$.

In this work, $x^* = \hat{x}_{j^*}$, the predicted optimal using the most probable model, j^*, based on the Bayesian probability of Eqn. (7). Our rationale is that by focusing on $\hat{x}_{j^*}$, one can obtain a better prediction around that point. Specifically, the P-optimal algorithm tries different experimental points by adding an additional row to

X_{j^*}, and finds the experimental point where $\sigma^2_{j^*}(\hat{x}_{j^*})$ is minimized. By performing an experiment at the point that minimizes $\sigma^2_{j^*}(\hat{x}_{j^*})$, the new experimental data will add confidence to the prediction $\hat{y}_{j^*}(\hat{x}_{j^*})$. One could use all j models and weight their predictions by P_j to compute the expected value of $\hat{x}$ and $\hat{y}(\hat{x})$, but if there are multiple local minimum, the resulting intermediate points might lead to very poor designs. Therefore, in the case study for this experimental design, the most probable model is always used to design the next experiment.

As part of our methodology development we also consider another new feature, which we call the grid algorithm. Its purpose is to further concentrate the experiments near $\hat{x}_{j^*}$. This is critical when partial models are being used, and is not necessarily achieved by D-, G-, or P-optimal, depending on the model or models being considered. The grid algorithm can be divided into five steps:

1. Define a set of grid points, x_g, in the experimental domain, Z.

2. Calculate a threshold value, $H(\hat{x}_{j^*})$:

$$H(\hat{x}_{j^*}) = f(\hat{y}_{j^*}(\hat{x}_{j^*})) + CI_{j^*}(\hat{x}_{j^*}) \tag{30}$$

3. Restrict consideration to grid points with outputs that may be below the threshold value, $f[\hat{y}_{j^*}(x_g)] - CI(x_g) < H(\hat{x}_{j^*})$. (These are the potential optimal points, at the α confidence level.)

4. Calculate the output of the experimental design function, $f_{ED}(x)$, at these remaining grid points and pick the best point, $x_{e,o}$, from this set.

5. Find the optimal experimental point, x_e, using $x_{e,o}$ as a starting point, constraining the search within a box centered at $x_{e,o}$, with sides equal to twice the grid spacing.

Eqn. (30) represents the case where the objective is to find the minimum value of $f(x)$. The lower bound of a prediction at point x is: $LB(x) = f(\hat{y}_{j^*}(x)) - CI(x)$

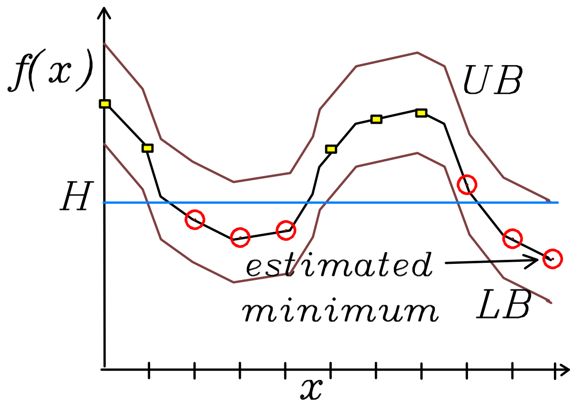


Figure 21: Graphical representation of the grid algorithm. Filled circles are potential minima, while open squares are not.

and the upper bound is: $UB(x) = f(\hat{y}_{j^*}(x)) + CI(x)$. In this case the confidence interval is added to the optimal value because one does not want to eliminate values of x in which $LB(x) < UB(\hat{x}_{j^*})$. This is shown graphically in Figure 21, in which all values of x having $LB < H$ are potentially the minimum (marked by circles). The points marked by squares are not potential minimum at a confidence level of α. Similar calculations can be performed when the objective is to find the maximum value. In this work, only two experimental parameters are explored, making the grid method easy to implement. For higher dimensions of the experimental parameters, a full gridding of the design space may not be practical. In such cases, a stochastic sampling could alternatively be used to obtain a sampling of the system over the experimental space in the regions of interest. To get parameters that fit better in the region of interest, one may also use the grid algorithm to limit parameter fitting only to points that are below $H(\hat{x}_{j^*})$.

CHAPTER IV

SIMULATION STUDY OF BEST EXPERIMENTAL DESIGN

Here we generate the "experimental" data using simulations of the process we are trying to model. By using simulated data, we are able to assess the performance of our experimental design methodology, and to quickly explore the effects of various parameters on its performance, including: the number of repetitions, the noise level, and the initial experiments performed. The insight gained here will be used when we apply our methodology in our experimental CVD system.

4.1 Initial Experimental Design Case Study

This case study was developed to test the initial experimental design methodology presented in Section 3.3. All model discriminations, graphs, and simulations were performed using Matlab 7 (R14) Service Pack 2 on a Dell computer with a Pentium III processor. Maximizing the grain size (minimizing nucleation density) is the "performance measure" and implies picking a new experimental setting which minimizes the grain size. In equation form

$$f(x) = \frac{1}{\hat{y}(x)} \tag{31}$$

and the full equation

$$\bar{D}_{m,n}(\mathbf{x}) = \left(\frac{P_m}{\hat{y}_m(\mathbf{x})} + \frac{P_n}{\hat{y}_n(\mathbf{x})}\right) \frac{[\hat{y}_m(\mathbf{x}) - \hat{y}_n(\mathbf{x})]^2}{2\sigma^2(\mathbf{x}) + \sigma_m^2(\mathbf{x}) + \sigma_n^2(\mathbf{x})} \tag{32}$$

In this case study, we are trying to pick the best of four models that output nucleation density, given *simulated* experimental data. The models are shown in Table 6 The first model is a constant which fits the average of the experimental data.

Table 6: Models used in initial experimental design case study

model	model form
constant	$N_{isl} = a$
polynomial	$N_{isl} = a + b \times T + c \times F$
mechanistic 1	$\frac{dN_1}{dt} = F(1-\kappa) - (\rho+1)K_{nuc}(\eta, T, E_i, E_d, N_1) - K_{agg}(\eta, T, E_i, E_d)$
	$\frac{dN_{isl}}{dt} = K_{nuc}(\eta, T, E_i, N_1)$
mechanistic 2	$\frac{dN_1}{dt} = F(1-\kappa) - (\rho+1)K_{nuc}(\eta, T, E_i, N_1) - K_{agg}(\eta, T, E_i)$
	$\frac{dN_{isl}}{dt} = K_{nuc}(\eta, T, E_i, N_1)$
i[1ex]	

The second model is a polynomial where N_{isl} is nucleation density, a, b, and c are fitted parameters, T is deposition temperature, and F is the flux of atoms to the surface of the substrate. The third model is mechanistic where N_1 is the density of isolated atoms on the surface. The fraction of the surface covered is represented by κ and is set to 0.15 monolayers at which point no further nucleation is expected [43]. ρ is the number of adatoms needed for a stable cluster and is set to the typical value of 2. K_{nuc} is the nucleation rate and K_{agg} is the aggregation rate [43]. In this mechanistic model, E_i (binding energy), E_d (diffusion activation energy), and η (capture number) are the fitted parameters.

The fourth model is the mechanistic model using a constant value for the E_d parameter (two fitted parameters instead of three). The set of differential equations are integrated by the Matlab function `ode23t` to predict the final value of the island density, N_{isl}, at κ monolayers (ML) of deposition and depending on the value of the flux, F.

The simulated experimental data are generated using the mechanistic model, using parameters as in Table 7. The study will have Gaussian random noise added to the data, of zero mean and with the magnitude given in Table 7. The purpose is to see how well the discrimination function works with multiple models. The noise was chosen to be 1×10^{-5} which is large enough to add variance to the data but small enough

Table 7: Parameters for simulated experimental data for the discrimination case study.

Parameter	value	units
E_i	0	eV
E_d	0.8	eV
ρ	0.2	–
repetitions of each data point	10	–
noise	1e-5	$\frac{\text{islands}}{\text{adsorption site}}$
T	873	K
	1073	K
F	0.1	$\frac{ML}{min}$
	100	$\frac{ML}{min}$

Table 8: Results of discrimination case study

models	error	$\bar{D}_{j,1}$	$\bar{D}_{j,2}$	$\bar{D}_{j,3}$	$\bar{D}_{j,4}$
1	2.0×10^{-7}	–	9.2×10^{-9}	1.0×10^{-4}	$\mathbf{1.5}\times10^{-4}$
2	2.6×10^{-8}	–	–	2.9×10^{-5}	4.2×10^{-5}
3	6.1×10^{-11}	–	–	–	1.5×10^{-9}
4	6.1×10^{-11}	–	–	–	–

whereas distinct model parameters could be distinguished. The initial simulated data for the study came from a 2^2 factorial experiment using the high and low settings for T and F given in Table 7.

The results for the study are shown in Table 8. The table shows the value of the discrimination function between the respective models and the error between the model and the experimental data. This was done as a pairwise comparison for all combinations of the models realizing that $\bar{D}_{j,k} = \bar{D}_{k,j}$. The experimental design will pick the largest $\bar{D}_{j,k}$ value and pick that for the next experiment. From the table the largest $\bar{D}$ is $\bar{D}_{1,4}$, where Model 1 is the least probable (highest error) and model 4 is the most probable. When the goal is to find the optimal operating point for a process, it does not make much sense to use the worst process model when designing subsequent experiments. Also, the experimental points where the two models disagreed the most, were not near the optimal point for the process.

4.2 Description of Revised Experimental Design Case Studies

In the previous section we learned that model discrimination was not designing experiments to quickly find the optimal point of the process, but was finding the next experiment to best determine which of the models was best. Since the goal was to quickly find the optimal point of a process, a revised methodology was developed and was presented in Section 3.4. To test the revised methodology two case studies are presented.

In the following two case studies are performed. The first case study is used to test the method on a simple system that is linear in the parameters. The second case study is used to test the method on a system of technological interest and uses several candidate models. Three models are purely empirical, and one model is mechanistic. This case study was chosen to reflect the ongoing experiments in our research group dealing with a chemical vapor deposition process [148, 150].

The grid algorithm was implemented using D-optimal, P-optimal, and a random selection of experiments. First, the D-optimal method is used to sequentially design the experiments, and parameter fits are made based on these experiments. At each iteration, the most probable model is used to predict the next experimental point. The entire methodology simulation is performed one hundred times at each noise level and the results are averaged. This process is then repeated using P-optimal, and then using a random selection of experiments. In the case of random selection of experiments, the grid algorithm first creates the list of possible optimal points. A point from this list is then selected randomly as the next experiment.

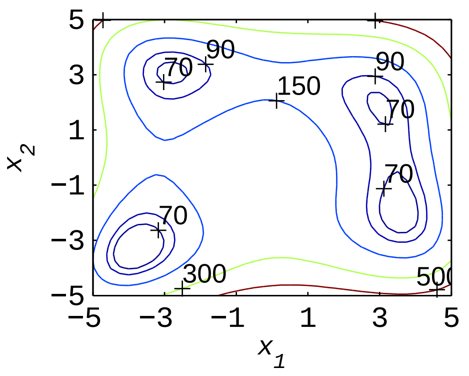


Figure 22: Modified Himmelblau function over $x_1 \in [-5,5]$ and $x_2 \in [-5,5]$

4.3 MHF case study

We first evaluate the method using the modified Himmelblau function (MHF)[32]:

$$y^{MHF}(x) = x_1^4 + x_2^4 - 21x_1^2 + 2x_1^2x_2 + 2x_1x_2^2 - 13x_2^2 - 13x_1 - 19x_2 + 227 \qquad (33)$$

We consider a range of [-5,5] for both x_1 and x_2. In this case study, $f(x)$ is the same as $y(x)$. The minimum of Eqn. (33) is at the point $x_1 = -3.80$ and $x_2 = -3.32$, with a corresponding function value of 43.3. The other local minima have function values of 63.5, 54.9, and 65.9. A contour plot of this function is shown in Figure 22.

This function is chosen because it possesses desirable properties for evaluating the experimental design method, including having a minimum in the interior and having multiple local minima. Eqn. (33) is used here to generate the simulated data, while a slightly different model is used to fit the sampled data:

$$\hat{y}_1(x) = x_1^4 + x_2^4 - 21x_1^2 + 2x_1^2x_2 + \theta_1x_1x_2^2 - 13x_2^2 + \theta_2x_2 + \theta_3 \qquad (34)$$

Eqn. (34) is missing the linear x_1 term from the original MHF function to create model mismatch, such that no values of the three fitted parameters (θ_{1-3}) will completely match all points. No model is ever perfect, so in practice there will always be some mismatch. Experimental design methods must be robust to these modeling errors.

The initial experiments for this case study are chosen to be a 2^2 factorial design (1 experiment performed at each corner of the design space) plus an experiment at the center of the design space with two repetitions, for a total of six experiments ($2^2+2\times(0,0)$). The center point is needed due to the nonlinear nature of the system being modeled, since there are quadratic terms in the MHF. The noise added (γ) to $y^{MHF}(x)$ is Gaussian and zero-mean, with standard deviations of 3, 15, or 30. The desired confidence interval (ϵ_{CI}) on the prediction is set to 5. The initial experiments are run, $\hat{\theta}_j$ is calculated, and the initial Bayesian probability is calculated using Eqn. (7), as done in existing methods.

The MHF case study is used to examine three of the main choices affecting the D- and P-optimal and the grid method designs. First, the effect of the initial experiments is compared using our original experimental design ($2^2 + 2 \times (0,0)$) and the experimental design from the Latin Hypercube [40]. Second, the size of the grid for the grid algorithm at three different levels is compared. Third, the effect of the tolerance on the stopping criteria is investigated.

4.4 Film growth case study

This case study is based on a chemical vapor deposition process. In CVD, a thin film is grown on a heated substrate by flowing an evaporated metalorganic precursor over the substrate [34]. CVD experiments and film characterization can be time consuming and costly, making CVD a good candidate for experimental design. Thin films grow after an initial nucleation of clusters on the substrate surface, and the density of nucleation sites on the substrate influences the film morphology and properties [43]. The purpose of this case study is to evaluate our experimental design method using multiple partial models. In this case study, the method picks the best of four possible models that predict the flux of atoms to the substrate surface, given the *simulated* experimental data. The two experimental settings considered are $x = [T, C]$. T is the deposition temperature, and C is the concentration of precursor in the reactor. The flux of precursor to the substrate is F, and the true flux of the system is simulated

Table 9: Constants for the film growth case study

Constant	Value	Units	Description
κ	0.15	—	Monolayers [43]
ρ	2	—	Number of adatoms per stable cluster
R	8.3145e-3	$\frac{kJ}{mol*K}$	Gas constant
C_{max}	1.5	$\frac{\mu mol}{L}$	Maximum C
T_{max}	1073	K	Maximum T
u	5.061	$\frac{L}{min}$	Flow of inert gas
A_{sub}	5.067e14	nm^2	Area of substrate
Ω	4.488e22	$\frac{nm^3}{mol}$	Molar volume
h_{ML}	0.2651	$\frac{nm}{ML}$	Thickness of a monolayer
E_a	19.25	$\frac{kJ}{mol}$	Activation energy for flux
E_i	0	eV	Binding energy
E_d	0.8	eV	Diffusion activation energy
η	0.2	—	Capture number

using

$$F = F_0 e^{\frac{-E_a}{RT}} \tag{35}$$

where F_0 is the flux as $T \to \infty$ and E_a is the activation energy for the flux. F_0 is calculated as

$$F_0 = \frac{C\Omega u}{A_{sub} h_{ML}} \tag{36}$$

where C is the concentration, u is the flow of inert gas, Ω is the molar volume of yttria, A_{sub} is the area of the substrate, and h_{ML} is the height of a monolayer of atoms [151]. F_0 is a function of C, making F a function of T and C. E_a was calculated assuming a flux of 300 $\frac{ML}{min}$ [43], at the maximum temperature $T_{max} = 1073\ K$, and maximum concentration $C_{max} = 1.5\ \frac{\mu mol}{L}$. Inserting these quantities into Eqns. (35) and (36), one obtains $E_a = 19.25\ \frac{kJ}{mol}$. The value of the constants in Eqns. (35) and (36) can be found in Table 9. The flux is then used to calculate the growth time for a film. This is calculated using a target film thickness of 150 nm divided by the flux in $\frac{nm}{min}$.

The four models we propose will calculate the flux of atoms to the substrate. Gaussian noise with zero mean and standard deviation in Table 10 is added to the flux simulated by Eqn (35) and (36). The first model is empirical with one fitted parameter:

$$\hat{y}_1(x) = \theta_1 \tag{37}$$

which is the average of the data when doing a least squares fit. The second model is also empirical but now with two fitted parameters

$$\hat{y}_2(x) = \theta_1 C + \theta_2 \tag{38}$$

which assumes there is no temperature dependence on the flux going to the substrate. The third model is empirical with three fitted parameters

$$\hat{y}_3(x) = \theta_1 + \theta_2 T + \theta_3 C \tag{39}$$

where the model does assume temperature and concentration dependence for flux. The fourth model is mechanistic and is similar to Eqn. (35) but with E_a as the fitted parameter and $F_0 = 10C$, calculated as

$$\hat{y}_4(x) = 10Ce^{\frac{-\theta_1}{RT}} \tag{40}$$

thus creating model mismatch from the true system.

The nucleation density is given by

$$\frac{dN_1}{dt} = F(1-\kappa) - (\rho+1)K_{nuc}(\eta, T, E_i, E_d, N_1) - K_{agg}(\eta, T, E_i, E_d) \tag{41}$$

$$\frac{dN_{isl}}{dt} = K_{nuc}(\eta, T, E_i, N_1) \tag{42}$$

and is based on mechanistic principles [43]. The set of differential equations are integrated by the Matlab function `ode23t` to predict the final value of the island density, N_{isl}, at κ monolayers (ML) of deposition and depending on the value of the flux, F, where N_1 is the density of isolated atoms on the surface. The fraction of the

Table 10: Parameters for the simulated experimental data for nucleation study in Eqns. (41) and (42).

	Parameter	Value	Description
	ν_e	1	Repetitions
	γ	5, 10, 20 $\frac{ML}{min}$	Gaussian standard deviation of flux
	ϵ_{CI}	1 $\frac{ML}{min}$	Desired confidence interval for nucleation study
	ϵ_{tol}	0.1	Tolerance for change in MSE
	t_{goal}	3 minutes	Growth time objective
	N_{goal}	10^{-3} $\frac{\text{islands}}{\text{adsorption site}}$	Nucleation density objective
x	T	873 K 1073 K	Low T setting High T setting
	C	0.3 $\frac{\mu mol}{L}$ 1.5 $\frac{\mu mol}{L}$	Low C setting High C setting

surface covered is represented by κ and is set to 0.15 monolayers at which point no further nucleation is expected [43]. ρ is the number of adatoms needed for a stable cluster and is set to the typical value of 2. K_{nuc} is the nucleation rate and K_{agg} is the aggregation rate [43]. There is no mismatch in this portion of the model.

The objective function for the film growth study is

$$min_x(f_{j^*}(x)) = (t(\hat{y}_{j^*}(x)) - t_{goal})^2 + (10^4(N_{isl}(x, \hat{y}_{j^*}(x)) - N_{goal}))^2 \qquad (43)$$

N_{goal} is the nucleation density desired for the film and t_{goal} is the desired growth time for the film. Values for both are shown in Table 10. This objective is chosen since chemical vapor deposition will usually be one of many steps in creating a product and such specifications would be given at the factory level. Thus, N_{goal} is a target on the quality, while t_{goal} is a target for the process. The confidence interval for the objective function is calculated using the confidence interval on the $t(\hat{y}_{j^*}(x))$ and $N_{isl}(\hat{y}_{j^*}(x))$ predictions based on the confidence interval for F. This is accomplished using absolute errors expanded to second order terms [30], due to the zero slope at the optima. Figure 23 is a contour plot of the objective function calculated using

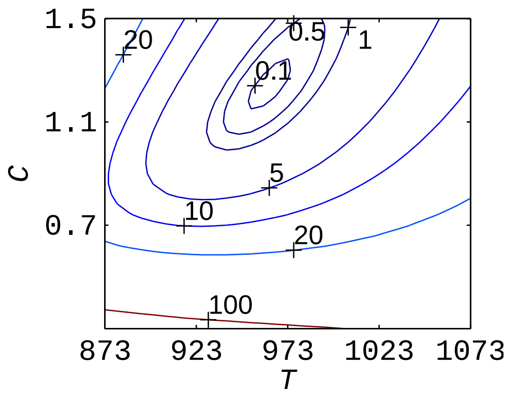


Figure 23: Contour plot of $f(x)$ for the film growth case study with no added noise.

Eqns. (35), (36), (41), (42), and (43).

The simulated experimental data are generated using Eqns. (35), (36), (41), and (42) using the parameters in Table 10. The study has zero mean Gaussian random noise (γ) added to the data, with the standard deviation given in Table 10. The initial simulated data are generated from a 2^2 factorial experiment using the high and low settings for T and C given in Table 10, with one repetition at each design point for a total of four experiments.

4.5 Results

All simulations were performed using Matlab Release 2007a, and all optimizations were carried out using Matlab's function `fmincon` for constrained optimization, which uses an SQP method.

4.5.1 MHF case study

The contour plot of the model with the best fit parameters of [1.48 -19.05 226.22] is shown in Figure 24. The model mismatch had the effect of eliminating two of the local optima from the experimental design space seen in Figure 22. However, the global minima location remains unchanged, but the value at the global optima is different. In this study equal importance has been put on finding the location of the optimal

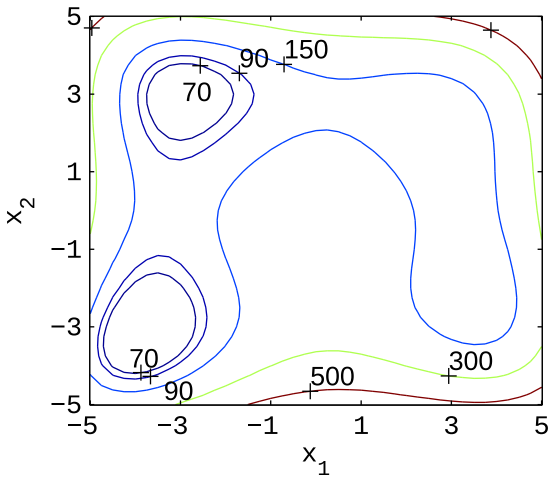
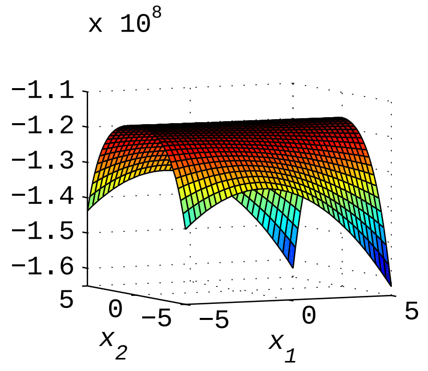


Figure 24: Contour plot of proposed model for MHF simulation study over $x_1 \in [-5, 5]$ and $x_2 \in [-5, 5]$ with the best fit parameters after the first iteration.

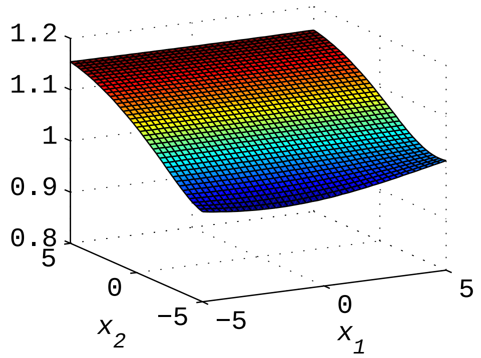
point, as well as predicting the true value at that location. Another case would be using a model which has a different global minimum than the true system.

Typical experimental design surfaces with respect to the parameters x_1 and x_2 are shown in Figures 25(a) and (b). The values on the vertical axis of the D-optimal surface are negative due to a negative sign in front of the optimality calculation, so that the function can be minimized instead of maximized. The D-optimal surface shows that the corners of the design space are all local minima for this design function. When designing an experiment using D-optimal and without the grid algorithm, experiments are designed only at these corner points, even though these points have already been run, or the corner points are not near the optimal point, or both. The P-optimal surface, on the other hand, has its minima not on the corners of the design space, but along the $x_2 = -5$ edge of the design space. Like D-optimal, P-optimal also tends to prescribe experiments on the boundaries, but the direction is influenced by the value of $\hat{x}$. Recall that in this case study $\bar{x}$ = [-3.80 -3.32].

The results are summarized in Tables 11–17. These tables are broken up into three different columns for each level of noise. Each of these columns is further split into three columns, one for each type of sequential experimental design function



(a) D-optimal vs. parameters for the MHF case study after the 1st iteration.



(b) P-optimal vs. parameters for the MHF case study after the 1st iteration.

Figure 25: Values of optimality functions in the MHF case study ($\gamma = 3$, 1st iteration) vs. parameters x_1 and x_2.

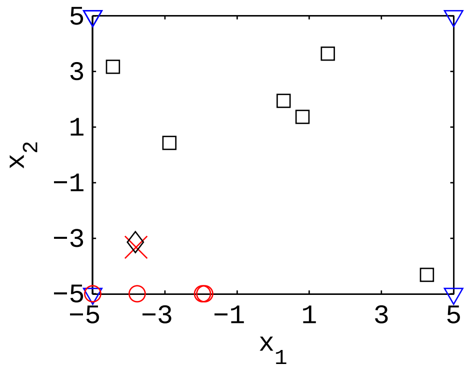
considered. The Rand columns are the results from sequential experiments sampled from a uniform distribution over the design space. Each table has nine rows which give the final values after the algorithm and experiments have terminated. The first row is $\hat{y}_{j^*}(\hat{x}_{j^*})$, from the most probable model j^* evaluated at $\hat{x}_{j^*}$. The second row is the prediction variance and the third row is the confidence interval, both at $\hat{x}_{j^*}$. The fourth row is the percentage error between $\hat{y}_{j^*}(\hat{x}_{j^*})$ and $y^{MHF}(\bar{x})$ which is represented by ζ. The fifth row is the mean squared error as calculated in Eqn. (8). The sixth row is the estimated model variance as calculated in Eqn. (28) and the seventh row is the average number of iterations. The eighth and ninth row contain the percentage error in $\hat{x}_{j^*}$ compared to $\bar{x}$. For this case study, $\hat{x}$ never matches $\bar{x}$ =[-3.80 -3.32] exactly,

Table 11: MHF case study with $2^2 + 2 \times (0,0)$ initial experiments using no grid, $\epsilon_{tol} = 0.1$.

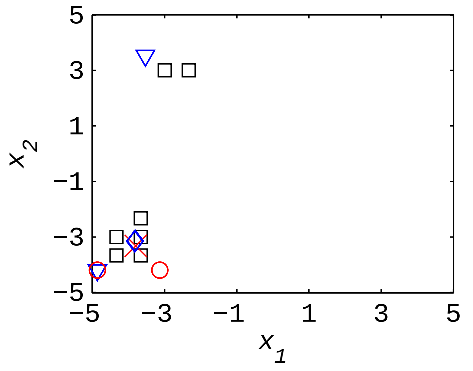
	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	14.7	14.7	16.4	15.1	15.4	16.5	15.2	15.5	12.0
$\sigma_{j^*}(\hat{x}_{j^*})^2$	1.8	1.2	65.5	35.5	26.6	138	153	94.0	278
$CI(\hat{x}_{j^*})$	2.9	2.4	16.5	12.2	10.7	24.0	25.4	20.0	34.3
ζ (%)	66	66	62	65	64	62	65	64	72
MSE_{j^*}	5.7	5.3	285	170	172	558	692	641	1080
$\hat{\sigma}^2$	8.9	8.3	375	234	244	741	963	894	1480
iter	2.3	2.1	6.1	5.3	4.7	6.1	5.2	4.8	5.3
x_1, % error	0.8	0.8	0.7	0.8	0.7	0.3	0.1	0.6	0.8
x_2, % error	4.1	4.1	4.4	4.1	4.0	4.3	9.6	5.7	17.0

which is the effect of the model mismatch. With some of the terms missing from the model in Eqn. (34), compared to Eqn. (33), the model's minima may not ever go through the true optimal point, no matter what experiments are performed. This is akin to the ideal gas law predicting a different state than the virial equation of state, which includes intermolecular forces that the ideal gas law does not. Throughout this case study, only designs at low noise are able to meet the goal of $CI(\hat{x}_{j^*}) < 5$, while most stop because the MSE stopped changing within $\epsilon_{tol} = 10\%$.

In Table 11, each column is an average of 100 simulations. One can see that without the grid algorithm, none of the experimental design methods leads to good performance at any noise level. The error in $\hat{y}_{j^*}(\hat{x}_{j^*})$ is greater than 60% in all noise levels. Another problem is at low γ, the CI for D- and P-optimal are extremely misleading. With such a small CI, an experimenter may think his model prediction is very good. This small CI can be traced back to the small MSE. Figure 26 shows experiments from different experimental design methods after the initial experiments have been run. Comparing Figures 22 and 26(a), D- and P-optimal sample along the edges of the design space without the grid algorithm, where $y^{MHF}(x)$ has very different values than $y^{MHF}(\bar{x})$. This leads to a poor fit at $\hat{y}_{j^*}(\hat{x}_{j^*})$, but a small MSE based on the sampled points.



(a) MHF case study experiments without grid algorithm



(b) MHF case study experiments with grid algorithm

Figure 26: Experiments from the different experimental design methods are marked: D-optimal ($\bigtriangledown$), P-optimal ($\circ$), Random ($\square$), and greedy ($\diamond$). The greedy experimental design samples at $\hat{x}_{j^*}$ only. $\times$ marks the true optimal point.

Table 12: MHF case study with 2^2+2$\times(0,0)$ initial experiments using δ=15, $\epsilon_{tol} = 0.1$.

	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	25.8	24.9	27.9	24.0	24.4	30.6	19.3	21.6	24.2
$\sigma_{j^*}(\hat{x}_{j^*})^2$	19.9	14.2	50.6	46.3	31.1	83.3	151	112	235
$CI(\hat{x}_{j^*})$	9.2	7.9	14.6	14.0	11.5	18.8	25.4	21.8	31.5
ζ (%)	40	42	36	45	44	29	55	50	44
MSE_{j^*}	113	77.7	250	231	205	376	727	736	955
$\hat{\sigma}^2$	154	113	337	322	287	519	1020	1030	1320
iter	5.3	3.8	5.8	4.9	4.8	5.2	4.7	4.8	5.2
x_1, % error	0.6	0.6	0.6	0.6	0.6	0.6	0.0	0.2	1.1
x_2, % error	4.6	4.8	4.8	4.5	4.8	4.7	9.7	8.1	19.1

Wanting to sample closer to $\bar{x}$, we implement the grid algorithm. In Figure 26(b), one can see that the grid algorithm does cause sampling closer to $\bar{x}$. The experiments that are not near $\bar{x}$ are instead close to another local minimum. A strength of the grid algorithm is the ability to search other potential local optima to find the best optimum in the experimental region. If the initial experiments lead to an incorrect $\hat{x}$, the method will not prohibit sampling at other potential optima.

In Table 12, one can see that the grid algorithm does lead to an improvement ($\zeta \approx 40\%$ in the low noise case instead of $\approx 60\%$), although the results still are not very good. Surprisingly, the random experiments with the grid method seem to lead to the best predictions of $\hat{y}_{j^*}(\hat{x}_{j^*})$. In this case, the grid algorithm identifies the potential optima, and the next experiment is chosen at random from these points. D- and P-optimal, on the other hand, use the potential optima from the grid algorithm, and then design the experiment at the point with the best respective value of the design function. Experimental designs like D- and P-optimal tend to maximize the variance of x, which becomes counterproductive when we are dealing with partial models. Unless the experiments are designed to sample $\hat{x}$, $\hat{x}$ is usually not the point chosen by the experimental design for the next experiment. The random method also samples at a larger variety of experimental points around the optima, whereas the D- and P-optimal methods converge to the same points in subsequent iterations. Note we are not prohibiting any of the designs from repeating previous experiments, and the figures do not explicitly indicate repeated experiments. If one designs the D- and P-optimal algorithms to always pick different points, the predictions from those optimalities might be improved.

Since the random experiments work better with the grid algorithm, and we would like to improve the predictions of $\hat{y}_{j^*}(\hat{x}_{j^*})$ even further, we investigate whether changing our initial experiments to a more random design will improve the results. The initial experiments are run using a Latin Hypercube design as the initial set of six

Table 13: MHF case study with Latin Hypercube initial experiments using $\delta = 15$, $\epsilon_{tol} = 0.1$.

	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	18.6	20.2	27.0	17.2	19.4	26.1	15.1	19.2	22.8
$\sigma_{j^*}(\hat{x}_{j^*})^2$	106	69.2	219	153	108	382	220	201	488
$CI(\hat{x}_{j^*})$	21.5	17.3	30.6	25.7	21.6	40.5	30.7	29.4	45.7
ζ (%)	57	53	37	60	55	40	65	56	47
MSE_{j^*}	429	382	572	623	577	799	962	1030	1130
$\hat{\sigma}^2$	622	549	820	908	836	1150	1370	1470	1590
iter	4.0	4.0	4.4	3.9	4.0	4.3	4.4	4.3	4.6
x_1, % error	0.3	1.2	2.4	0.1	1.0	4.5	1.5	2.8	7.2
x_2, % error	5.3	16.7	23.1	8.8	14.8	27.1	19.6	29.2	30.6

Table 14: MHF case study with $2^2+2\times(0,0)$ initial experiments using δ=5, $\epsilon_{tol} = 0.1$.

	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	14.8	14.8	21.2	18.8	20.3	25.8	17.4	20.0	23.4
$\sigma_{j^*}(\hat{x}_{j^*})^2$	2.4	2.2	69.5	42.8	50.8	84.3	136	111	225
$CI(\hat{x}_{j^*})$	3.3	3.2	17.1	13.5	14.7	18.9	24.1	21.7	30.9
ζ (%)	66	66	51	57	53	40	60	54	46
MSE_{j^*}	5.5	5.1	271	186	333	343	666	712	819
$\hat{\sigma}^2$	8.4	7.8	372	263	457	471	929	1000	1140
iter	2.4	2.5	5.1	4.5	5.2	5.2	4.9	4.9	4.9
x_1, % error	0.8	0.8	0.7	0.7	0.7	0.7	0.0	0.3	0.9
x_2, % error	4.1	4.1	4.4	4.4	4.2	4.4	9.7	11.8	17.2

experiments. Comparing Tables 12 and 13 one can see that randomizing the initial experiments does not have the desired effect of improving $\hat{y}_{j^*}(\hat{x}_{j^*})$, although one does obtain results in fewer iterations. Randomizing the initial experiments has the effect of being too space-filling. The CI are larger and more accuracte but this has the effect of including too many potential optima in the grid, effectively expanding the sampling area which the grid algorithm is designed to reduce. One also sees increased $\bar{x}$ error, especially in x_2. If one refers to Eqn. (34), one sees that x_1 and x_2 are not completely independent, which may be why the error is concentrated on x_2, and not distributed equally between both. Since the LHS method does not provide a significant benefit, we proceed with our initial $2^2 + 2\times$(0,0) initial design.

Table 15: MHF case study with 2^2+2×$(0,0)$ initial experiments using δ=30, $\epsilon_{tol} = 0.1$.

	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	26.3	26.5	31.3	23.1	23.6	29.7	19.3	19.8	23.9
$\sigma_{j^*}(\hat{x}_{j^*})^2$	21.1	13.7	38.0	49.2	32.8	79.9	127	110	214
$CI(\hat{x}_{j^*})$	9.5	7.7	12.6	14.4	11.8	18.3	23.1	21.7	30.2
ζ (%)	39	39	28	47	45	31	55	54	45
MSE_{j^*}	115	84.1	198	246	212	385	690	664	864
$\hat{\sigma}^2$	157	119	268	344	297	524	947	938	1200
iter	5.2	4.5	5.7	5.0	4.8	5.5	5.4	4.6	4.9
x_1, % error	0.6	0.5	0.6	0.6	0.6	0.6	0.4	0.8	1.3
x_2, % error	4.6	5.0	4.8	4.5	4.8	4.6	6.3	15.8	19.4

Table 16: MHF case study with $2^2 + 2 \times (0,0)$ initial experiments using $\delta = 15$ and $\epsilon_{tol} = 0.05$

	$\gamma = 3$			$\gamma = 15$			$\gamma = 30$		
	D	P	Rand	D	P	Rand	D	P	Rand
$\hat{y}_{j^*}(\hat{x}_{j^*})$	28.9	26.9	31.5	25.7	24.2	34.5	21.7	21.8	32.0
$\sigma_{j^*}(\hat{x}_{j^*})^2$	11.9	9.1	39.0	30.5	21.0	55.6	93.1	68.4	161
$CI(\hat{x}_{j^*})$	7.0	6.1	12.6	11.1	9.2	15.0	19.4	16.7	25.5
ζ (%)	33	38	27	41	44	20	50	50	26
MSE_{j^*}	110	76.5	253	267	222	378	774	723	996
$\hat{\sigma}^2$	137	101	324	330	280	475	951	911	1250
iter	10.1	7.5	8.3	10.5	9.8	10.0	10.9	9.7	9.7
x_1, % error	0.6	0.6	0.5	0.5	0.5	0.3	0.3	0.1	1.2
x_2, % error	4.8	4.8	4.9	4.7	4.9	6.8	6.5	10.2	17.9

One of the parameters of the grid algorithm is the number of points on the grid of the design space, δ, which could be varied to improve $\hat{y}_{j^*}(\hat{x}_{j^*})$. In Table 14, the results for $\delta = 5$ are presented. In comparing this data with that of Table 12 where $\delta = 15$, one can see that too coarse a grid will lead to poorer model predictions, but still better results than without the grid algorithm, especially at higher noise levels. With a coarse grid and high noise, the constrained search area may also contain many local optima, which hinders the search for the true optimal point. A fine grid, $\delta = 30$, leads to improved predictions, but takes significantly more computational time. For this work, $\delta = 15$ is deemed a good compromise.

A comparison of different ϵ_{tol} was performed without a grid as well as with $\delta = 15$

Table 17: MHF case study with $2^2 + 2 \times (0, 0)$ initial experiments and "greedy" experimental design, $\epsilon_{tol} = 0.1$.

	$\gamma = 3$	$\gamma = 15$	$\gamma = 30$
$\hat{y}_{j^*}(\hat{x}_{j^*})$	33.6	37.0	35.7
$\sigma_{j^*}(\hat{x}_{j^*})^2$	24.0	37.3	111
$CI(\hat{x}_{j^*})$	10.0	12.3	21.2
ζ (%)	22.5	14.7	17.7
MSE_{j^*}	168	342	940
$\hat{\sigma}^2$	225	427	1180
iter	6.1	10.3	10.0
x_1, % error	0.48	0.01	1.48
x_2, % error	5.25	9.06	20.3

(Table 16). The lowered ϵ_{tol} leads to better predictions from the model and tighter CI when comparing Tables 12 and 16, but also more iterations, as one might expect. The better performance is due to the increased amount of data around the optimal point, approximately two times as much as when $\epsilon_{tol} = 0.1$. If the objective of the experiments is to get an accurate model prediction, a smaller tolerance (i.e. willingness to run more experiments) will cause a better fit. However, if the emphasis is on quickly finding the optimal point, then a larger ϵ_{tol} may produce adequate results. At any point during the design phase, one may choose to terminate the experiments, before reaching the prespecified tolerance.

Since sampling near the optimum point seems to improve $\hat{y}_{j^*}(\hat{x}_{j^*})$, why not sample at the predicted optimum point? We implemented this using a "greedy" method and the results are shown in Table 17 and marked by a $\diamond$ in Figure 26(b). Rather than minimize a separate experimental design function, this method chooses the estimated optimal point of the model j^* as the next experimental point to be sampled. This leads to similar predictions and number of iterations to Table 15 for the random plus grid algorithm for low noise, while for high noise the performance and iterations are similar to random plus grid algorithm with Table 16. These similarities point to the importance of the stopping criterion for the experimental design. The grid algorithm can obtain similar results to the greedy method given the right stopping criterion, but

Table 18: Film growth case study with 2^2 initial experiments using no grid, $\epsilon_{tol} = 0.1$.

	$\gamma = 5$			$\gamma = 10$			$\gamma = 20$		
	D	P	Rand	D	P	Rand	D	P	Rand
$f_{j^*}(\hat{x})$	0.05	9×10^{-9}	0.02	0.07	5×10^{-8}	0.03	0.02	0.01	0.04
$\sigma_{j^*}(\hat{x}_{j^*})^2$	221	10.8	26.8	304	141	108	754	637	515
$CI(\hat{x}_{j^*})$	7.40	1.95	2.80	9.03	5.24	5.46	16.1	13.6	9.93
MSE_{j^*}	169	7.1	8.48	255	125	33.6	676	579	138
$\hat{\sigma}^2$	675	28.4	30.6	907	401	115	2190	1800	434
iter	8.1	6.7	8.8	8.4	7.2	9.0	7.7	7.4	9.4
$\hat{T}$, % error	1.54	0.47	0.02	2.91	0.46	0.2	1.43	0.05	0.03
$\hat{C}$, % error	9.19	2.85	0.19	17.25	2.82	1.49	8.70	0.35	0.03

has the advantage of exploring more of the experimental region than if one were to use the greedy method. The better model prediction of greedy makes sense as more experiments are run and experiments are located at the optimal point. The least squares method will then focus on minimizing the error from these points, making the fit to $y(\bar{x})$ better. This "greedy" case can be viewed as the limit of the grid algorithm with $\delta \to \infty$ and $\alpha \to 100\%$. Ultimately, the performance in this case study is limited by the model mismatch between Eqns. (33) and (34). One could probably increase the prediction accuracy more by leaving out the initial experiments during the parameter fitting, which would cause the fit to be better around the more recently sampled points.

4.5.2 Film growth case study

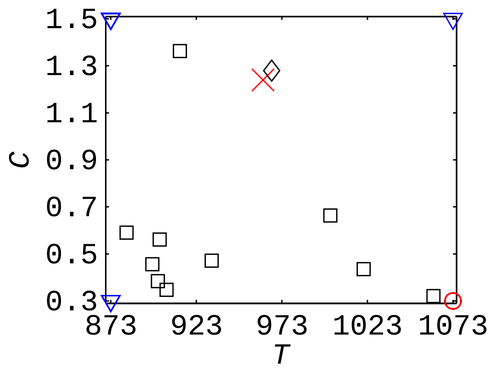
The film growth case study is different from the first case study because four candidate models are now used. The first case study always underpredicted the optimum but in this case study the models are able to predict the desired output of the process ($t = 3$ min. and $N_{isl} = 10^{-3}$); however $\hat{x}_{j^*}$ is not necessarily $\bar{x}$. For this reason we restricted the models to only fit data that falls below $H(\hat{x})$, or at least enough data to make X be full rank. This will increase the confidence interval on the prediction due to fitting to fewer points, but ultimately should enable more accurate fits at the

Table 19: Film growth case study with 2^2 initial experiments grid=15, $\epsilon_{tol} = 0.1$.

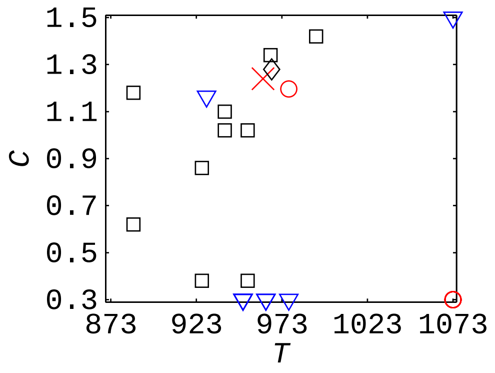
	$\gamma = 5$			$\gamma = 10$			$\gamma = 20$		
	D	P	Rand	D	P	Rand	D	P	Rand
$f_{j^*}(\hat{x})$	0.12	2×10^{-5}	2×10^{-8}	0.03	2×10^{-3}	2×10^{-8}	0.03	6×10^{-4}	2×10^{-3}
$\sigma_{j^*}(\hat{x}_{j^*})^2$	392	106	21.3	331	150	67.6	885	738	845
$CI(\hat{x}_{j^*})$	16.1	3.98	2.64	11.9	6.11	4.45	18.0	15.4	10.4
MSE_{j^*}	227	65.8	12.5	248	130	37.3	774	677	133
$\hat{\sigma}^2$	933	223	43.1	894	436	120	2440	2110	396
iter	7.4	7.4	8.4	7.3	7.5	9.0	7.5	7.5	9.2
$\hat{T}$, % error	0.03	0.15	0.14	0.31	0.30	0.05	1.19	0.07	0.03
$\hat{C}$, % error	0.36	0.88	1.53	2.13	1.81	0.77	7.47	0.70	0.41

optimal point. The true optimum in this case is at $\bar{T} = 962\ K$ and $\bar{C} = 1.24\ \frac{\mu mol}{min}$. The tables in this section show the objective function for the most probable model, instead of the output, $y(x)$, since $f(x) \neq y(x)$. The $CI(\hat{x}_{j^*})$ shown is also for $f(x)$ and not on $y(x)$ for the same reason.

In Table 18, the third empirical model (Eqn. (39)) is the most probable in two thirds of the simulations, the second empirical model (Eqn. (38)) or the mechanistic model (Eqn. (40)) were the next most probable in equal proportion, and the first empirical model is picked as most probable the least often of the four. Here, $f_{j^*}(\hat{x})$ are quite small ($f_{j^*}(\hat{x}) < 0.1$) and the $CI(\hat{x}_{j^*})$ are quite large compared to size of the objective function at the optimal point. In the no grid case, all three experimental designs minimize $f_{j^*}(\hat{x})$ well at all noise levels. The random experimental design has the smallest error for predicting $\hat{T}$ and $\hat{C}$. In Figure 27(a), one can see the experiments for each experimental design. The experiments for D- and P-optimal are on the corners of the design region, far away from the optimal point. The random design samples all over the design region leading to the better $\hat{x}$ prediction. In Table 19 the results of the film growth experiments using the grid algorithm are shown. At first glance, one would say that the grid did not bring any improvement to the experiments at all. In fact, the $CI(\hat{x}_{j^*})$ are not significantly different than without



(a) Film growth case study experiments without grid algorithm.



(b) Film growth case study experiments with grid algorithm.

Figure 27: Film growth study experiments from the different experimental design methods are marked: D-optimal ($\bigtriangledown$), P-optimal ($\circ$), and Random ($\square$), and greedy ($\diamond$). The greedy experimental design samples at $\hat{x}_{j^*}$ only. $\times$ marks the true optimal point, with γ =10 s, $\epsilon_{tol} = 0.1$, and $\delta = 15$.

the grid. However, the % error in $\hat{T}$ and $\hat{C}$ is improved with the grid for D- and P-optimal. One can see the effect of the grid on the experiments in Figure 27(b) where there are more experiments around the optimal point than without the grid. In Figure 28 the possible optimal grid points are plotted for sequential experiments 1, 4, and 7 for one simulation of the random experimental design. At each successive experiment, the range of possible optimal points shrinks. It is also worth noting that in iterations 1 and 4, the grid retains the lower left corner of the experimental area. Here, the confidence interval for these points is so large that they are included as possible optima.

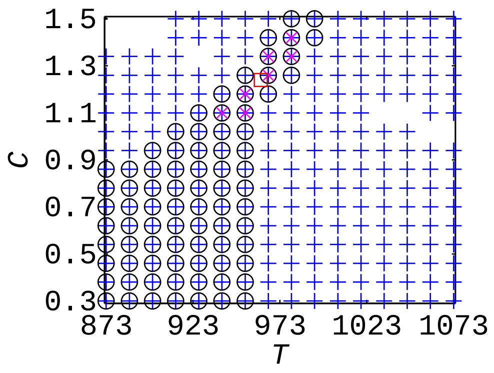


Figure 28: Possible optimal points after grid algorithm using the random experimental design. First sequential experiment marked by '+', fourth sequential experiment marked by '∘', and seventh seqential experiment marked by 'x'. The true optimal point is [962 1.24] and is marked by '□'.

One interesting effect of the grid algorithm is in the resulting most probable model for each experimental design. Using the grid algorithm, the most probable model is still the third empirical model, but the mechanistic model is the second most likely, and the first empirical model is never considered the most probable model as it sometimes was without the grid. The grid had the effect of eliminating one of the possible models, and clearly identified a temperature dependence in the data (most probable models both included T) that was not apparent without the grid. The mechanistic model does well with better sampling of the experimental area around the optimal point, but still not better than Eqn. (39). This indicates the mechanistic model may need modification for improved prediction, but not necessarily that a whole new model is needed to explain the data.

In Table 20 are the results of the film growth experiments using the greedy experimental design where only the optimal point is used as the next experiment. In this case study, the greedy experimental design does not yield significant improvements to the objective function, CI, or finding the optimal point. However, the greedy experimental design picks the mechanistic model as the most probable model rather than any of the empirical models. Again, the greedy method does do quite well in

Table 20: Film growth case study with 2^2 initial experiments and the "greedy" experimental design, $\epsilon_{tol} = 0.1$.

	$\gamma = 5$	$\gamma = 10$	$\gamma = 20$
$f_{j^*}(\hat{x})$	1×10^{-8}	2×10^{-4}	6×10^{-3}
$\sigma_{j^*}(\hat{x}_{j^*})^2$	5.29	137	972
$CI(\hat{x}_{j^*})$	0.58	4.59	11.3
MSE_{j^*}	17.8	40.7	226
$\hat{\sigma}^2$	21.6	102	458
iter	8.8	11.6	11.4
$\hat{T}$, % error	0.08	0.42	0.53
$\hat{C}$, % error	0.07	2.92	3.89

most simulations, but in some simulations $\hat{x}$ is incorrect and sampling at the same incorrect point repeatedly does not improve $\hat{x}$. Good performance is achieved using the grid algorithm or the random sampling. Based on these case studies, combining the two together will provide good estimates of the optimal point while still allowing for some exploration and model building.

4.6 Conclusions

In conclusion, the P-optimal and grid methods of experimental design were introduced. The grid method in the MHF case study used in conjunction with other experimental designs result in enhanced performance of these designs, with better model predictions (15-20% better than without the grid algorithm) and more accurate confidence intervals which contain the true optimum of the MHF model (43.3). The film growth case study achieved good results using random experimental design; however the grid algorithm did lead to better sampling of the experimental region, better $\hat{x}$ prediction, and led the experimenter to a dependence on T that would have been missed otherwise. As with any experimental design method, it is up to the experimenter to define the objective for the experiment. In this work the main objective is to obtain good model predictions of $\bar{x}$ of a process for use in process design. The grid algorithm is a useful tool because it balances the trade-off between an optimal design such as D-optimal and a “greedy” experimental design. It also allows for exploring other regions with potential optima rather than focusing on one optima which may be erroneous. The methodology presented here combines existing experimental design approaches for empirical models with those for mechanistic models. Thus, the experimenter does not have to decide ahead of time whether to use empirical only, or mechanistic only, since both may be useful. As with many experimental design techniques, there is not one setting for the grid algorithm or stopping criteria that works for all experiments. The experimenter needs to combine his knowledge of the system being studied and interpretation of the available data to come up with a viable experimental design.

CHAPTER V

EXPERIMENTAL STUDY USING CVD REACTOR TESTBED

In Chapter 4 we showed how the experimental design would perform with various parameters for the algorithm. In this way we could compare the results and select an appropriate design to be used on our experimental system. We acquired information from the simulation studies to select the size of the grid, and an idea of how our desired confidence level influences the results of the experimental design algorithm. We determined that a random experimental design used in conjunction with the grid algorithm would help us more quickly find the optimal point for our experimental chemical vapor deposition (CVD) system.

5.1 Initial Experiments

There are many experimental parameters that could be studied for this system. Substrate type, O_2 flow rate and/or its ratio to the argon flow rate, total gas flow rate through the reactor, pressure, deposition temperature, and precursor molar flow rate are just a few of the adjustable parameters for the system. From previous work on this system, we determined that deposition temperature and precursor molar flow rate would be the most influential to begin our study. The high and low settings for deposition temperature and precursor molar flow rate are shown in Table 21.

The high temperature is limited by the button heater used to heat the substrate, which has burned out several times in the past four years. The voltage limit for the heater is 22 volts under vacuum, so we decided to limit ourselves to no higher than 19 volts. Since the wafer temperature reached by the heater given an applied voltage

Table 21: Experimental high and low settings for the experimental study.

Setting	High	Low	units
T	775	600	^{o}C
$\dot{n}$	100	200	$\frac{\mu mol}{min}$

varies slightly from day to day, 775^{o}C was chosen as the high temperature and 600^{o}C was chosen as the low temperature, which correspond to approximately 18.5 V and 13 V respectively. The low setting was determined by minimum temperatures used to deposit yttria seen in other research [132].

The high precursor molar flow rate was chosen given the temperature of the oven and the limit on the amount of precursor detectable by the UV system. If the oven temperature is too high and the integration time on the UV detector is too low, there will be too much precursor in the line such that the detector will saturate. One may be able to use time-normalized experiments from the Ocean Optics OOIBASE32 software, but this feature is not incorporated into the LabVIEW program and would therefore be incompatible with the PI controller for the precursor molar flow rate. The lower limit for molar flow rate was selected as half the upper limit, since $\dot{n}$ is obviously required for film growth. It was desirable for the deposition rate to be fast enough to complete a deposition in a day, so the process settings were chosen to still enable a fast deposition rate. What was sacrificed in raising the deposition rate was a loss in robustness for growth time repeatability as discussed in Chapter 6.

The goal of this study is to find the temperature and precursor molar flow rate that minimize the following objective function

$$\min_x(f_{j^*}(x)) = ((RMS(x, \hat{y}_{j^*}(x)) - RMS_{goal}))^2 \tag{44}$$

The film roughness RMS is the experimentally measured quantity

$$RMS = \sigma_h \tag{45}$$

where σ_h is the standard deviation of the height measurements from the AFM image.

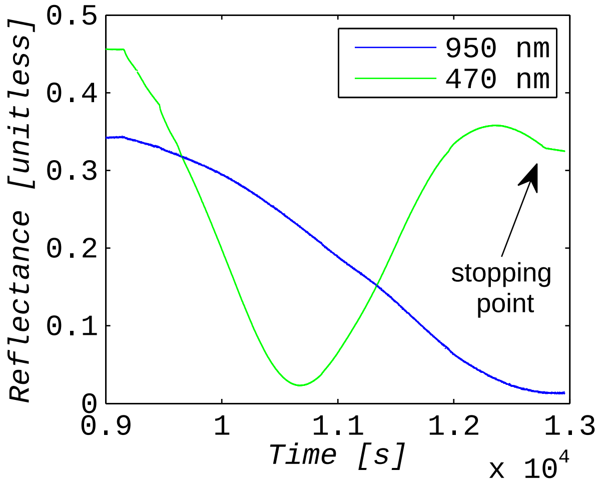


Figure 29: Reflectometer data for Sample 1.

The target roughness of $RMS_{goal} = 7$ nm is our desired property of the processed film. The film roughness is measured using the atomic force microscope (AFM) over a 2×2 μm area of the wafer surface. Three different locations on the wafer are measured and one of the locations is measured twice to account for any scan-to-scan variability. The RMS roughness is reported using the "statistical quantities" in the Gwyddion program after the raw data has been flattened using a polynomial of degree 3. The roughness of a thin film is often measured by AFM [101, 96] and is a robust metric from the AFM (see Chapter 6).

All of the films are grown to approximately the same thickness as measured by the *in situ* reflectometer. The target thickness for the films is 120 nm and is measured *ex situ* using ellipsometry. The reflectometer data for Sample 1 is shown in Figure 29. In particular we used the data from the 470 nm wavelength as a measure of the film thickness since this wavelength oscillates more as the film grows. One can make a qualitative comparison of growth rates by comparing the reflectometer data from the different experiments. A fast growth rate will have a short period in the 470 nm reflectance while a slower growth rate will have a much longer period, assuming the refractive index of the film is constant [21].

Table 22: Experimental measurements for the initial experiments.

Setting [units]	1	2	3	4	5	6
Pressure [torr]	1.6	1.6	1.6	1.6	1.6	1.6
Dummy argon [sccm]	200	200	200	200	200	200
Carrier argon [sccm]	148–160	32–104	95–117	117–137	79–122	82–103
Precursor argon [sccm]	40–48	89–160	81–95	63–78	81–113	97–118
Precursor molar flow[$\frac{\mu mol}{min}$]	99	200	101	205	201	202
O_2 [sccm]	50	50	50	50	50	50
Main oven T [^{o}C]	155	155	155	155	155	155
Evaporator T [^{o}C]	141	140	142	137	140	140
Voltage [V]	12.9	18.5	18.4	13.2	13.4	12.6
Current [A]	6.15	8.13	8.16	6.25	6.29	6.11
Heater T [^{o}C]	697	976	965	717	718	680
Wafer T [^{o}C]	610	770	782	607	605	610

Similar to the simulation study from Chapter 4, the initial experiments will be a 2^2 factorial experiment, but each point is not repeated as in the simulation study. Once the initial experiments are run some repetition experiments are performed, but are typically not run for every experimental setting [12, 90]. The repetitions are necessary to estimate the variability inherent in the process to differentiate the variability caused by changing the process settings. Here the low T/high $\dot{n}$ point is chosen for the repeated experiments. The experimental settings for the initial experiments are shown in Table 22, and the results of the initial experiments are shown in Table 23. Experiments 5 and 6 are repetitions of Experiment 4.

In Table 22, one can also see the benefit of the PI controller on the molar flow rate of the precursor. The flowrates for the carrier argon and precursor argon change significantly throughout each experiment. This reflects the changing rate of evaporation in the evaporator, which the PI controller corrects for by changing the flowrates of the carrier and precursor argon, while maintaining a total flow rate of 200 sccm throughout the experiment. With the changing gas flowrates, the precursor molar flowrate as monitored by the UV sensor was kept very close to the setpoints of 100 and 200 $\frac{\mu mol}{min}$. The total argon flow through the reactor was selected as 200 sccm to ensure that there was enough argon to reach the required molar flow despite variations in the absorbance from the UV sensor.

There are two pressure sensors on the equipment each with different ranges. The first has a range of atomospheric pressure to 1 torr, while the second has a range of 10 torr to 10 mtorr. The second pressure sensor is more precise at lower pressures, so the pressure recorded in Table 22 is from the second pressure sensor. The oxygen flow was set to 50 sccm to ensure there was an excess of oxygen for the precursor to react. An experiment with 150 sccm of oxygen was performed to make sure oxygen was not the limiting reagent for deposition. The deposition rate and the roughness of the resulting film with an increased flow of oxygen was unchanged from oxygen at 50 sccm, so 50 sccm was used for the rest of the experiments. The effect of oxygen flow may have been hidden by another interaction between experimental settings. A full screening experiment would be required to conclusively discount oxygen's effect on the thin film deposition.

In Table 24 one sees the thicknesses of the films as reported by ellipsometry, along with the fitting parameters used in the model. The mean square error refers to the error between the fitted model and the acquired optical data. A_n is a fitted parameter in the Cauchy model used to fit the data and to calculate the refractive index of the film, which for Y_2O_3 is nominally 1.9 [94]. The roughness layer was included in the model to reduce the mean square error but was set at 20 nm instead of adding it as an additional fitted parameter. This roughness does not correspond exactly with the roughness reported from AFM data. AFM software calculates the roughness using the root mean square of height irregularities. In spectroscopic ellipsometry, an effective medium model (EMM) is used. In EMM, the roughness is modeled as the thickness of an effective layer based on optical responses measured by the ellipsometer. Some work has been done to quantify the difference between the two models. Some have reported that the EMM is twice the AFM thickness [26, 71, 98] while others

Table 23: Experimental results from initial experiments.

Experiment	Scaled T	Scaled $\dot{n}$	Roughness	T [^{o}C]	$\dot{n}$ [$\frac{\mu mol}{min}$]
1	-1	-1	10.3 ± 0.6	610	99
2	1	1	5.6 ± 0.5	770	200
3	1	-1	6.3 ± 0.5	782	101
4	-1	1	8.8 ± 1.2	607	205
5	-1	1	12.2 ± 1.5	605	201
6	-1	1	11.6 ± 0.5	610	202

Table 24: Thickness of films determined via ellipsometry averaged over six samples and parameters used in the Cauchy model.

Experiment	Thickness	A_n	roughness	MSE
1	125 ± 2	1.71	20	40
2	119 ± 4	1.88	20	16
3	134 ± 3	1.88	20	15
4	136 ± 5	1.76	20	26
5	123 ± 2	1.77	20	29
6	128 ± 3	1.78	20	23

have come to the conclusion that it is impossible to predict the correlation [45]. Based on measurements of roughness in Table 23, the correlation between the two measurements is approximately two. A fit of the model to experimental data is shown in Chapter 2 as Figure 12. Also, 20 nm is extremely small to be measured optically using a wavelength ¿ 400 nm with any accuracy.

All AFM images were processed by flattening the image using a polynomial of degree three in the Gwyddion software and the RMS roughness was calculated from the image. The roughness of the initial experiments are shown in Table 23 and are averaged over four images at three different locations on the film. This was done to estimate the variability of roughness due to location on the film and due to variability from one scan to the next. One can see in the image the different grains on the film surface. The grains are not of uniform size, but vary in size and shape from the random deposition process. From Table 23 there are a few interesting points to highlight. First, the deposition temperature seems to have a large effect on the

Table 25: Estimated average main effects and interaction effects from the initial experiments.

Effect	Value
T	-4.86
$\dot{n}$	-0.02
$T\dot{n}$	-0.58

roughness of the film. A lower deposition temperature leads to rougher films while high deposition temperature films are much smoother. While the temperature has a large effect, the molar flow rate does not appear to influence the roughness of the films. Currently, the lowest standard deviation for a roughness measurement was 0.5 nm, which is large considering the process settings' effect on roughness of 4 nm. Optimizing the gain settings and other parameters of the AFM may improve this variation in the measurements and improve the accuracy of a roughness model.

The main effects and the interaction effects were estimated roughly using the equations (6-1), (6-2), and (6-3) from [84] using $n = 1$ (n in [84] is repetitions of experiments) and averaging Experiments 4, 5, and 6 ($mean = 10.86$) and using the mean as the result for the low T/high $\dot{n}$ result. The results are shown in Table 25. The estimated effects confirm the qualitative observations. The strongest effect is indeed from T, and effect of $\dot{n}$ is negligible. An interesting result is the $T\dot{n}$ interaction effect, which is small compared to the T effect but much larger than the $\dot{n}$ effect alone.

5.2 Models of the experimental data

With the analysis of the initial experiments complete, potential models of the process are needed for the experimental design methodology. Our goal is to predict the surface roughness of the thin film which is important to such film properties such as dieletric properties [5] and on the structure of multilayer films [29].

Seven potential roughness models are included in the experimental design. The empirical models are

$$\begin{aligned} RMS_1 &= \phi_{11} + \phi_{12}T && (46) \\ RMS_2 &= \phi_{21} + \phi_{22}T + \phi_{23}\dot{n} && (47) \\ RMS_3 &= \phi_{31} + \phi_{32}T + \phi_{33}T\dot{n} && (48) \\ RMS_4 &= \phi_{41} + \phi_{42}T\dot{n} && (49) \\ RMS_5 &= \phi_{51} && (50) \end{aligned}$$

The roughness empirical models all use scaled T and $\dot{n}$.

The hybrid models will be similar to the model in Chapter 4, except instead of using N_{isl} directly, it will be an input to Equations (51) and (52) to output film roughness.

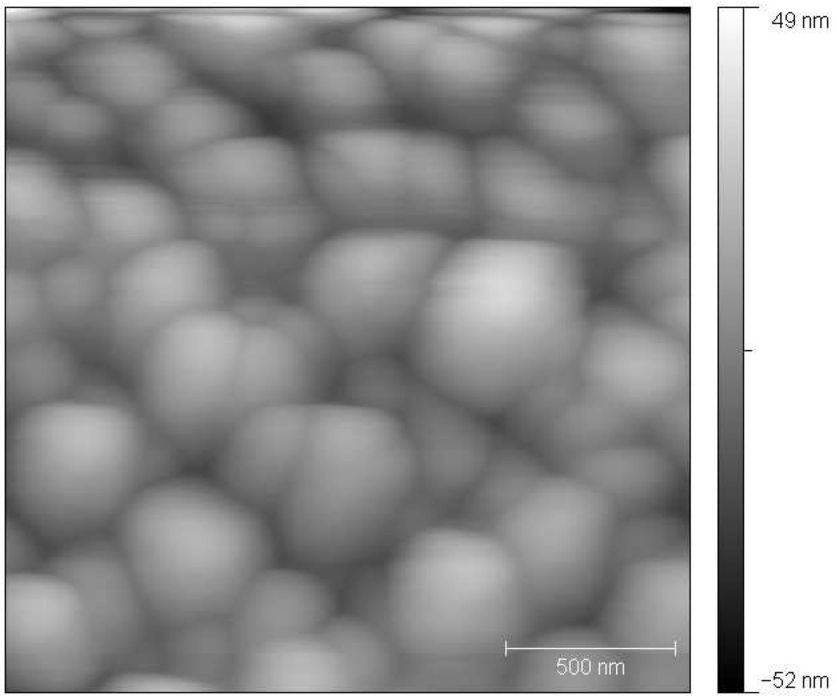
$$\begin{aligned} RMS_6 &= \phi_{61} + \phi_{62}\sqrt{N_{isl}} && (51) \\ RMS_7 &= \phi_{71}\sqrt{N_{isl}} && (52) \end{aligned}$$

Also, the flux, F for Equations (41) and (42) is calculated using Equation (53).

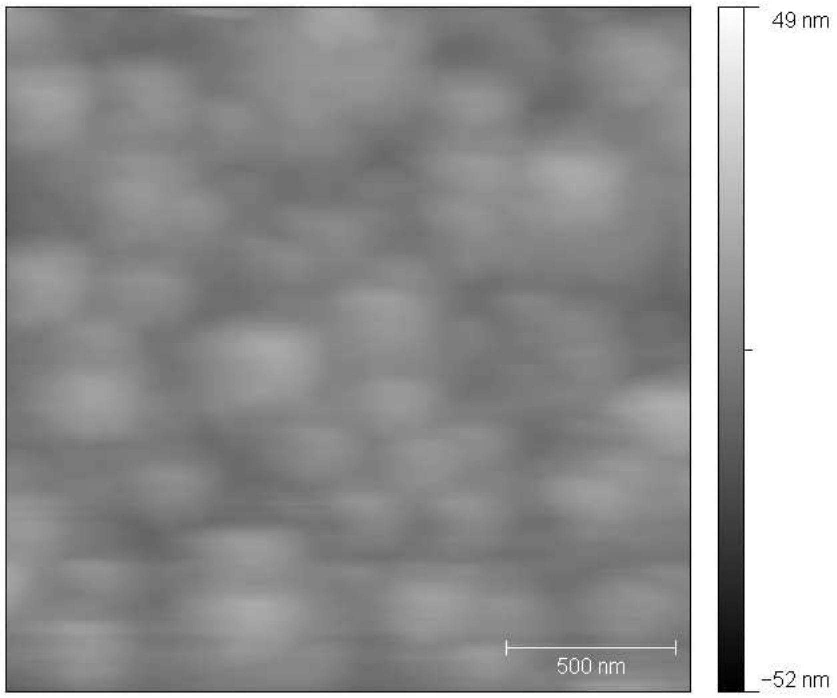
$$F = \omega\dot{n} \tag{53}$$

where $\omega = \frac{1\text{mol}}{10^6\mu\text{mol}}\Omega\frac{1}{h_{ML}}\frac{1}{A_{substrate}} = 3341.07\frac{ML}{\mu mol}$. The constants used in Equations (41) and (42) are in Table 26.

What the hybrid models are doing is drawing a relationship between nucleation density and roughness. Figure 30(a) is a film deposited at low T and Figure 31(a) is a film deposited at high T. These two films have very different roughnesses and qualitatively look very different. Figure 31(a) is smoother, but also appears to have a larger grain density than Figure 30(a). This makes sense when considering the nucleation density equations in Chapter 1, especially Equation (15). As T increases, β becomes smaller, causing N_ρ in Equation (16) to increase, which increases N_{isl} on the substrate surface. The many nuclei prohibit the atoms from congregating to only

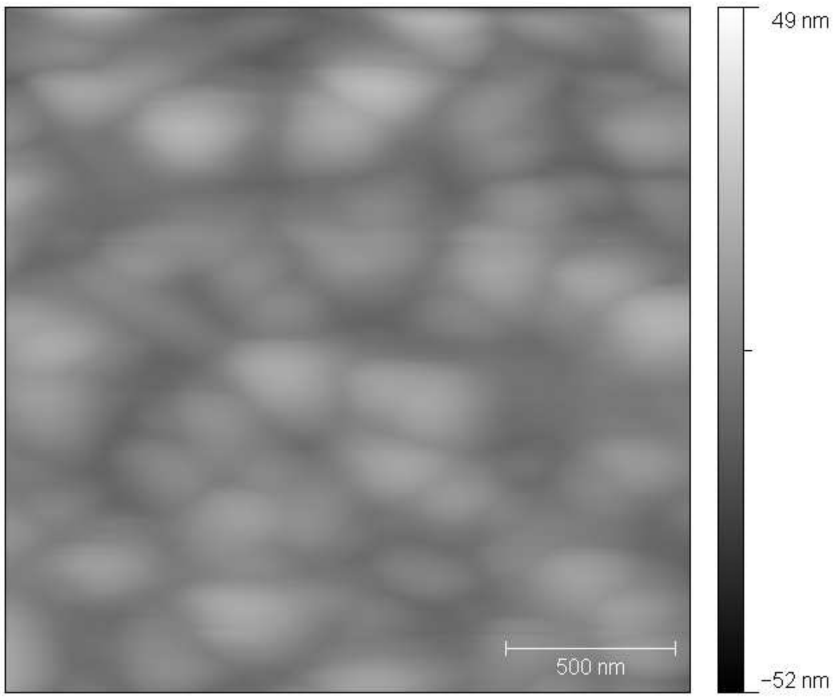


(a) AFM image of Experiment 1

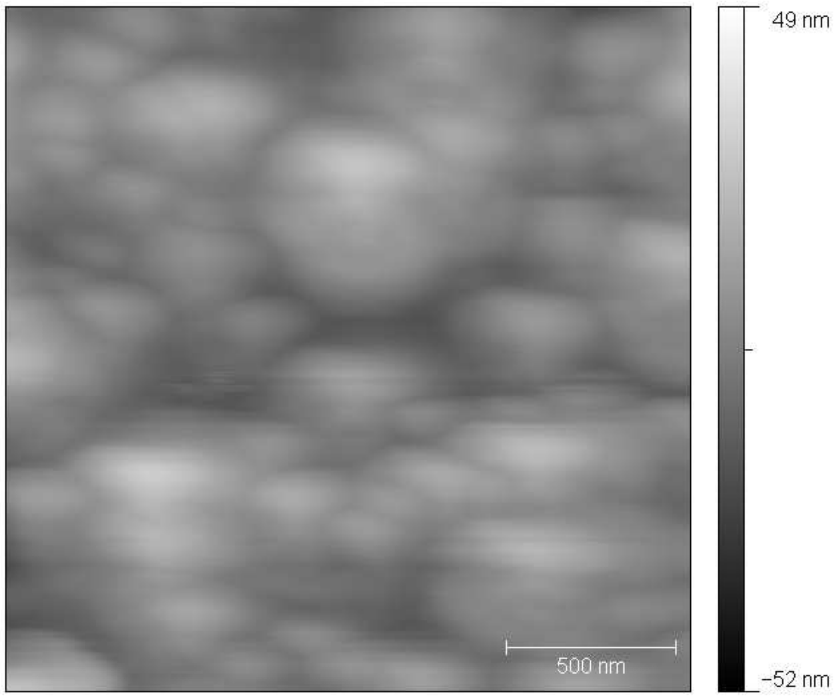


(b) AFM image of Experiment 2

Figure 30: AFM images of Experiments 1 and 2 to compare the effect of T on the thin film roughness.

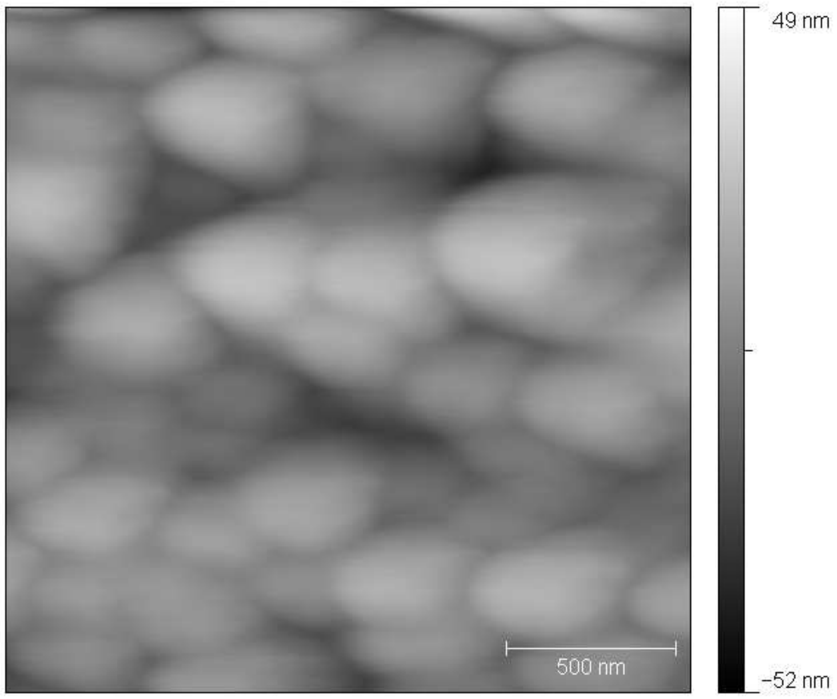


(a) AFM image of Experiment 3

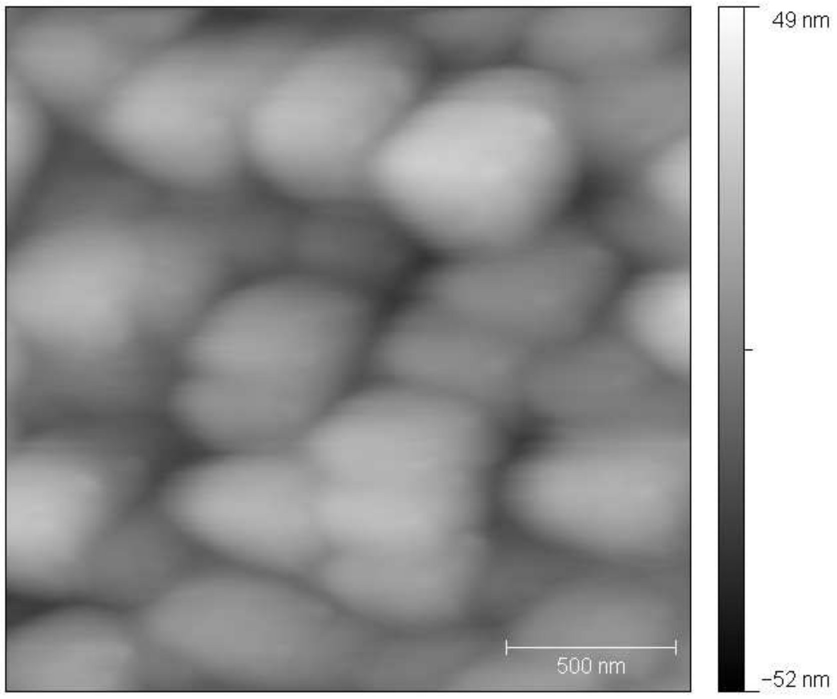


(b) AFM image of Experiment 4

Figure 31: AFM images of Experiments 3 and 4 to compare the effect of T on the thin film roughness.



(a) AFM image of Experiment 5



(b) AFM image of Experiment 6

Figure 32: AFM images of Experiments 5 and 6 to compare the effect of T on the thin film roughness.

Table 26: Constants for the film growth case study

Constant	Value	Units	Description
κ	0.15	—	Monolayers [43]
ρ	2	—	Number of adatoms per stable cluster
R	8.3145e-3	$\frac{kJ}{mol*K}$	Gas constant
Ω	4.488e22	$\frac{nm^3}{mol}$	Molar volume
h_{ML}	0.2651	$\frac{nm}{ML}$	Thickness of a monolayer
$A_{substrate}$	5.07×10^{14}	nm^2	Area of substrate
E_i	0	eV	Binding energy
E_d	0.8	eV	Diffusion activation energy
η	0.2	—	Capture number

Table 27: Results for roughness models after initial experiments

	form	MSE	P	ϕ_1	ϕ_2	ϕ_3
(46)	$\phi_{11} + \phi_{12}T$	1.17	**0.21**	8.31	-2.41	–
(47)	$\phi_{21} + \phi_{22}T + \phi_{23}\dot{n}$	1.17	0.15	8.30	-2.40	0.05
(48)	$\phi_{31} + \phi_{32}T + \phi_{33}T\dot{n}$	1.10	0.15	8.24	-2.34	-0.29
(49)	$\phi_{41} + \phi_{42}T\dot{n}$	5.66	0.10	8.83	-0.88	–
(50)	ϕ_{51}	6.34	0.13	9.11	–	–
(51)	$\phi_{61} + \phi_{62}\sqrt{N_{isl}}$	1.54	0.18	19.1	-104	–
(52)	$\phi_{71}\sqrt{N_{isl}}$	18.3	0.08	85.9	–	–

a few nuclei, causing the film to grow uniformly. The square root relationship between N_{isl} and roughness comes from the distance between grains which is $\approx \frac{1}{\sqrt{N_{isl}}}$.

In Table 27 are the parameter fits for the roughness models and their respective mean squared error and probability. Equation (46) was the most probable model after the initial experiments due to the strong dependence on T identified earlier, even though it did not have the lowest error of the seven models. The added constant in the hybrid model (51) made that model the second most probable and much more probable than (52), which did not have a fitted constant. All of the proposed models in Table 27 performed better than (50), which is simply the average of the available data, except for (52) which lacked a fitted constant term.

Equation (48) had the lowest error of all the models as expected from Table 25's result for the average effects from the initial experiments. Equation (48) includes both T and the $T\dot{n}$ interaction term in the model which are the terms with the largest calculated effects. Equations (46) and (47) have equal error, but (47) has a term for the $\dot{n}$ effect, therefore (46) is more probable of the two equations.

The results in Table 27 highlight an advantage of the new experimental design approach presented in Chapter 3. If one were doing a factorial design approach, after estimating the main effects, Equations (46) or (47) would be picked as the model and the rest of the experiments would be designed using one of these models. If a hybrid model approach was being taken, either Equation (51) or (52) would be chosen and further experiments would be designed to improve the estimates of the parameters of one of these models if using D-optimal design. In the approach here, multiple models are specifically included because no model is perfect. Additionally, Equation (50) is considered as a reality check to the experimenter. When Equation (50) has a probability greater than or comparable to the probabilities of the other models being considered, then a new relationship between the settings should be considered either for an improvement to the current models or for a new model altogether.

5.3 Sequential Experimental Design

After the initial experiments had been run, the results were put into the experimental design algorithm to see where the next experiment should be run. The Matlab code used for this can be found in Appendix D. The desired confidence interval for the most probable model was 1 nm and the change of MSE was set to be 0.1 for the stopping criterion. The scaled T, scaled $\dot{n}$, and roughness from Table 23 were given as inputs to the function.

To design the next experiment, the random experimental design with the grid algorithm was chosen since it was the best result from the simulation study. For

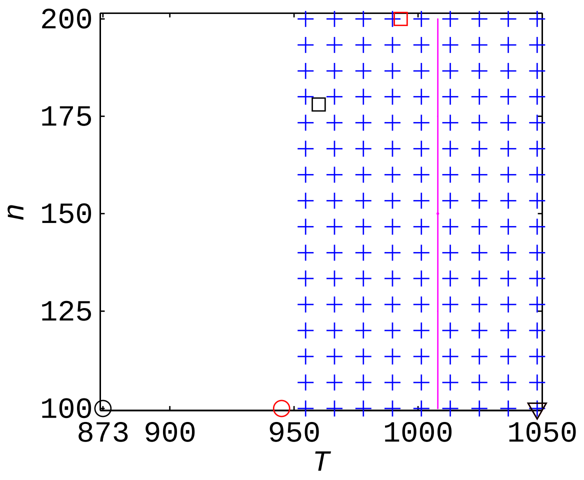


Figure 33: Points left in the grid after the initial experiments are shown as (+) plus the predicted optimal point (vertical line), the random experimental points (□), the P-optimal points (○), and the D-optimal points (▽). Experiments designed with the grid algorithm are shown in red.

comparison, the next experiment was also predicted using D- and P-optimal with and without the grid, as well as the random experimental design with and without the grid, and these different experimental points can be seen in Figure 33. The predicted optimal point after the first iteration is shown as a vertical line at the optimal temperature since the model shows no molar flow rate dependence, while the experimental points without the grid are black shapes, and with the grid are indicated by red shapes. Without the grid the D-optimal experiment is in the bottom right of the experimental design region. The predicted optimal is located on the T axis at $T = 1008$ K. The D-optimal design did not change with the grid and the experiment is predicted for the lower right corner of the design space. The P-optimal design is changed by the grid algorithm (GA) since the lower temperature settings are excluded from the experimental design space. The random experimental design point without the grid still occured within the grid space, which is not that unlikely since the grid space is large after the first iteration. These points are all on Figure 33 for comparison to the experiment that was actually run. Unlike the simulation study where nearly

Table 28: Experimental settings for the sequential experiments.

Setting	[units]	7	8
Pressure	torr	1.6	1.6
Dummy Argon	sccm	200	200
Carrier Argon	sccm	0–61	119–137
Precursor Argon	sccm	138–200	73–81
precursor molar flow	$\frac{\mu mol}{min}$	200	173
O_2	sccm	50	50
Main Oven Temperature	^{o}C	155	155
Evaporator Temperature	^{o}C	140	140
Voltage	V	16.5	16.8
Current	A	7.35	7.49
Heater Temperature	^{o}C	858	881
Wafer Temperature	^{o}C	718	741

unlimited "experiments" are possible, here these comparison points are not actually run experimentally due to the cost of experiments. The random selection with the grid algorithm was chosen as best from Chapter 4. The random experimental design with the grid is at $T = 720$ K and $\dot{n} = 200$ $\frac{\mu mol}{min}$ and that is Experiment 7 in Table 29. Experimental settings for the sequential experiments can be found in Table 28.

Since the most probable model is (46) which is linear in T, the $\dot{n}$ setting does not factor into the new experimental setting. This is seen clearly in Figure 33 in the grid space where the potential optimal points depend only on the T setting. From the analysis in the previous section, T was the most important factor, but the interaction between T and $\dot{n}$ was not negligible. This highlights a weakness in the optimal experimental designs where a lot of weight is put on the model for designing the next experiment. Equation (46) is the most probable in Table 27, but (51) is also quite probable, and should not be dismissed. By using a random experimental design rather than a D-optimal design, one can still explore the experimental design space to better define the relationships between the process settings and the process output while not putting too much confidence in any one model. For example, after running more experiments, Equation 47 could become the most probable model.

Table 29: Experimental results after initial experiments.

Experiment	Scaled T	Scaled $\dot{n}$	Roughness	T [^{o}C]	$\dot{n}$ [$\frac{\mu mol}{min}$]
7	0.37	1	8.5 ± 0.5	720	200
8	0.6	0.46	8.8 ± 0.7	750	173

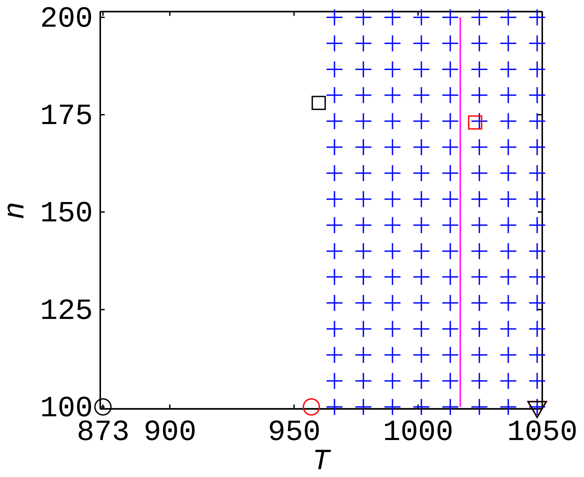


Figure 34: Points left in the grid after the first sequential experiment are shown as (+). The optimal point is a vertical line, the random experimental points are (□), the P-optimal points are (○), and the D-optimal points are (▽). Experiments designed with the grid algorithm are shown in red.

The results for the roughness models after the first sequential experiment are in Table 29 as Experiment 7. The parameter fits and model probabilities in Table 30 are relatively unchanged which indicates the experiment confirmed the previously observed trends but did not provide much additional insight into the process. Remember, however, that the goal of the study is not to find the best model (model discrimination) nor to get the best parameter fits for the models (D-optimal), but rather to find the optimal point of the process. Looking at Figure 34, information has been added about the location of the optimal point, as the optimal line has shifted.

The possible optimal points from the grid algorithm are displayed in Figure 34 as well as the predicted optimal point and the experimental design points from each of the methods. The grid space is slightly smaller, reflecting a smaller confidence

Table 30: Results for roughness models after first sequential experiment

	form	MSE	P	ϕ_1	ϕ_2	ϕ_3
(46)	$\phi_{11} + \phi_{12}T$	1.13	**0.21**	8.49	-2.30	–
(47)	$\phi_{21} + \phi_{22}T + \phi_{23}\dot{n}$	1.11	0.15	8.42	-2.27	0.17
(48)	$\phi_{31} + \phi_{32}T + \phi_{33}T\dot{n}$	1.11	0.15	8.46	-2.24	-0.19
(49)	$\phi_{41} + \phi_{42}T\dot{n}$	4.85	0.10	8.82	-0.88	–
(50)	ϕ_{51}	5.48	0.13	9.03	–	–
(51)	$\phi_{61} + \phi_{62}\sqrt{N_{isl}}$	1.47	0.18	18.7	-98.2	–
(52)	$\phi_{71}\sqrt{N_{isl}}$	15.9	0.08	84	–	–

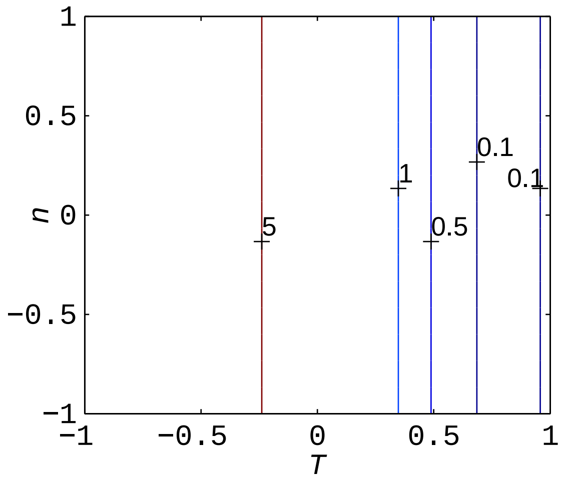


Figure 35: A contour plot of the objective function using Equation (46) based on Experiments 1–7.

interval due to more experiments. It is also interesting that the predicted optimal point has shifted with the new experiment. The contour plot of the objective function is shown in Figure 35. The right side of the figure has a valley where the minimum value of the objective function is located. With the added experiment the valley shifted along with the predicted optimal point. The predicted optimal point did move, but to a point that was another potential optima from the grid algorithm. The experimental points without the grid algorithm are shown again in black shapes. The next experiment from the random experimental design is T=1013 K and $\dot{n} = 173\frac{\mu mol}{min}$ and is Experiment 8 in Table 29.

After the second sequentially designed experiment, the results of the roughness

Table 31: Results for roughness models after the second sequential experiment

	form	MSE	P	ϕ_1	ϕ_2	ϕ_3
(46)	$\phi_{11} + \phi_{12}T$	1.27	**0.20**	8.73	-2.11	–
(47)	$\phi_{21} + \phi_{22}T + \phi_{23}\dot{n}$	1.24	0.14	8.64	-2.08	0.21
(48)	$\phi_{31} + \phi_{32}T + \phi_{33}T\dot{n}$	1.26	0.14	8.71	-2.06	-0.12
(49)	$\phi_{41} + \phi_{42}T\dot{n}$	4.25	0.11	8.85	-0.86	–
(50)	ϕ_{51}	4.80	0.15	9.00	–	–
(51)	$\phi_{61} + \phi_{62}\sqrt{N_{isl}}$	1.61	0.18	17.9	-87.8	–
(52)	$\phi_{71}\sqrt{N_{isl}}$	14.1	0.08	82.3	–	–

Table 32: Confidence intervals at the predicted optimal point for three models in the experimental study as well as the confidence interval on the objective function ($CI_{f(x)}$) for each model.

	(46)		(50)		(52)	
	CI_{RMS_1}	$CI_{f(x)}$	CI_{RMS_5}	$CI_{f(x)}$	CI_{RMS_6}	$CI_{f(x)}$
initial experiments	2.05	0.0004	2.89	5.956	2.39	0.0012
first iteration	1.71	0.0003	2.34	4.708	1.98	0.0019
second iteration	1.65	0.0003	1.96	3.917	1.91	0.0088

models in Table 31 are still relatively unchanged. Improvement has come in the form of a tighter prediction on the optimal point as can be seen in Table 32 for (46), (50), and (52). The CI for each of the models' predicted optimal point is reduced, and one becomes more certain where the true optimal point lies in the experimental space. Equation (46 is most useful for design because the confidence interval on the objective function is small. Equation (50) has a large $CI_{f(x)}$ because of its prediction at the predicted optimal point being far from the target of 7 nm.

The experimental design was carried out using the second sequential experiment and the grid points are shown in Figure 36. The predicted optimal point again moves along the valley depicted in Figure 35, and is located at $T = 1032$ K and $\dot{n} = 100\ \frac{\mu mol}{min}$. The next experiment from the random experimental design using GA is $T = 1036$ K and $\dot{n} = 153\ \frac{\mu mol}{min}$.

The stopping criterion for the experimental design were quite simple. Since the confidence interval on the measurements of roughness ranged between 0.5 and 1.5 nm, a goal confidence interval of 1 nm may be unattainable and meaningless if it is lower

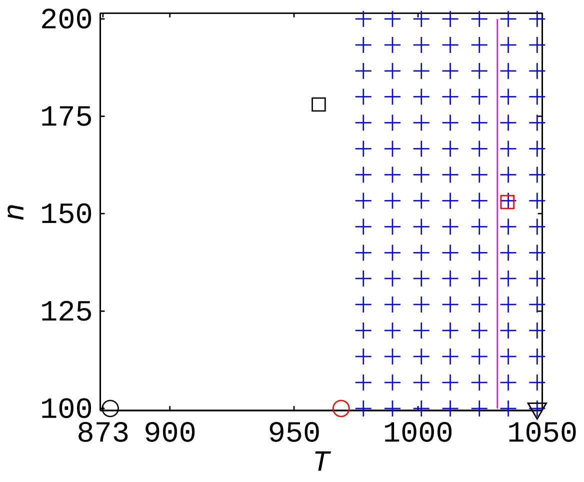


Figure 36: Points left in the grid after the second experiment are shown as (+). The optimal point is a vertical line, the random experimental points are (□), the P-optimal points are (○), and the D-optimal points are (▽). Experiments designed with the grid algorithm are shown in red.

than the measurement error. The change in MSE for each iteration was 0.03 and 0.12 respectively. The last experiment increased the MSE and the experimental design would continue. The MSE, however, only measures the error of the model prediction and does not indicate how well the model performs. For example, the MSE may be low for a model, but the observed trend of the data may not be accurately captured by the model.

The goal confidence interval is a nice guideline for when to stop experiments, but it does not accurately reflect the decision of the researcher. When the goal is to find the optimal point, the question is not how confident one is of the predicted optimal point, but how the confidence interval compares to the other potential optimal points. This is shown graphically in Figure 37. Point 2 is the predicted optimal point (assuming one is trying to find the minimum), but Point 1 may still be the true optimal point based on the overlap of the confidence intervals. It is still probable that the true value of Point 1 is less than Point 2. Depending on the differences in process settings between the two points, it may be profitable to run more experiments to distinguish between

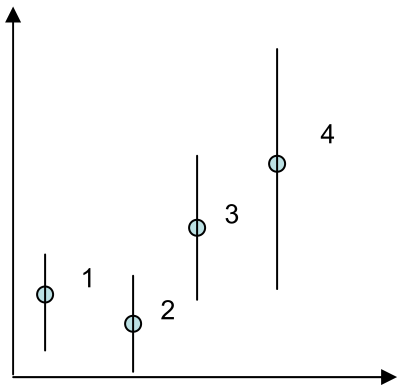


Figure 37: Four possible optimal points and their confidence intervals. Point 2 is the predicted optimal point (minimum) currently. Here, Point 1 may still be the true optimal point based on the overlap of confidence intervals, but Point 3 and Point 4 are probably not.

the two points. However, it is much less likely that Point 3 or 4 are the true optimal point. While they are still potential optimal points since their confidence intervals overlap Point 2's, there is a small probability that their values are less than Point 2's. A better stopping criterion may be one where there is a direct comparison of CI's of potential optimal points or calculating the probability that another experiment would signficantly change the location of the optimal point.

5.4 Conclusion

In this study we have evaluated the experimental design methodology using our CVD reactor testbed. The initial experiments were performed and seven roughness models were proposed and tested through sequential experimentation. The average T effect was found to be the largest for affecting the film roughness and the interaction effect of $T\dot{n}$ was also found to be important in modeling the roughness. The empirical model of Equation (46) was the most probable model throughout experimentation and the final predicted optimal point was found to be $T = 1032$ K and $\dot{n} = 100\ \frac{\mu mol}{min}$. This point is located in a valley of the objective function as depicted in Figure 35. With each successive experiment, the grid space was reduced as shown in Figure 38 reflecting a

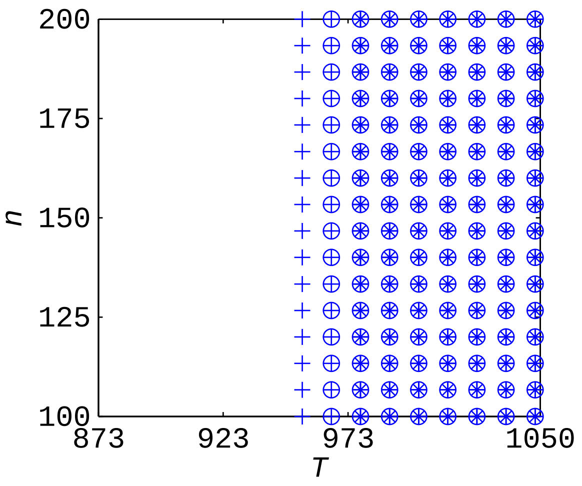


Figure 38: Points left in the grid after each experimental iteration. The first iteration points are marked by a '+', the second iteration points are marked by a (∘), and the third iteration grid points are marked as '*'.

smaller confidence interval on the predicted optimal point. Since the empirical model of Equation (46) was most probable of the proposed models, the existing hybrid models should probably be modified to include a stronger T dependence or another form of a hybrid model which specifically addresses the strong dependence on T should be proposed to predict roughness. Equation (51) in particular performed well, but may need modification to improve its performance over the other models. To further improve roughness models, an improvement in the AFM scans will be needed to decrease the variation from one scan to the next.

CHAPTER VI

LIMITATIONS OF EXPERIMENTAL DESIGN

In this chapter we will explore other aspects of experimental design that a researcher should be concerned about. The adequacy of a model should not be overlooked as one does not want to design experiments on the basis of a model that does not fit the experimental data very well. In our experimental design methodology, we explicitly define a worst-case model (a model that simply outputs the average of the experimental data). This model is incorporated into the design methodology to make sure the model we are fitting is useful for explaining the experimental data. This aspect is studied by fitting models to the growth time of films made using the CVD reactor testbed.

The second aspect of experimental design we are concerned with here is determining whether the measurement technique is appropriate to draw the necessary conclusions using experimental design. The choice of what to model is dependent on what can be deduced from the experiments. Trying to model a property which cannot be accurately measured is another pitfall experimenters may face. This is especially important when exploring on the nano scale where new techniques have become available to measure properties of thin films. To explore this we will try to model the grain size of thin films grown using the CVD reactor testbed.

6.1 Fitting the Growth Time of a Thin Film

The films for this section will be the same films used for the roughness study in Chapter 5. The experimental settings for each film are in Table 22. Similar to Chapter 5 we will be studying the effect of T and $\dot{n}$ and the high and low settings for each are shown in Table 21. The growth times for each of the initial six experiments

Table 33: Experimental results from initial experiments for growth time.

Experiment	Scaled T	Scaled $\dot{n}$	Growth time	T [^{o}C]	$\dot{n}$ [$\frac{\mu mol}{min}$]
1	-1	-1	60.1	600	100
2	1	1	82.1	775	200
3	1	-1	51	775	100
4	-1	1	32.2	600	200
5	-1	1	65	600	200
6	-1	1	95	600	200

Table 34: Estimated average main effects and interaction effects on the growth time from the initial experiments.

Effect	value
T	4.55
$\dot{n}$	17.65
$T\dot{n}$	13.45

are shown in Table 33.

From Table 33, one can see that the data is very noisy. Experiments 4, 5, and 6 are the repetition experiments and have a high variability which will make modeling the growth time very difficult if not impossible. However, for the purposes of this illustration, we will assume that the experimenter overlooks this and proceeds with his experimental design anyway. After running a factorial design the next step of a typical factorial experimental design is to estimate the average main and interaction effects of the process parameters being studied. Using the same method as in Chapter 5, the effects were estimated and are shown in Table 34. The values for each of the effects are large and the largest effect is $\dot{n}$, with the $T\dot{n}$ interaction effect also strongly present, and the T effect being the smallest of the three effects and the value is small relative to the observed growth times.

The estimated T effect is an interesting result since previous work has cited a linear increase or flat rate of reaction with increasing temperature [59, 103, 132]. This relationship depends on whether the reactor is in the reaction limited or diffusion limited regime. In the reaction limited regime, the surface deposition is limited by how quickly the surface reaction proceeds which is related to the substrate surface temperature. In the diffusion limited regime, the surface deposition is limited by how quickly the precursor can diffuse to the substrate surface. The temperature where it transitions from a reaction limited to diffusion limited reaction varies slightly from one CVD reactor to the next, but for Y_2O_3 is in the range of 580 to 600^{o}C [59, 103]. These studies did not specify how the substrate temperature was obtained meaning the temperatures reported were most likely recorded from a thermocouple on the back of the heater (corresponding to the "Heater Temperature" in Table 22). Even if we were to take the lower of the two reported temperatures (the wafer temperature), then we are still operating in the diffusion limited regime.

6.1.1 Models for Growth Time

Six potential growth time models were used to fit the available data. These models are all empirical in nature, unlike the mechanistic model that was used with the roughness models. The models are

$$t = \xi_{11} + \xi_{12}T \tag{54}$$

$$t = \xi_{21} + \xi_{22}T + \xi_{23}\dot{n} \tag{55}$$

$$t = \xi_{31}T + \xi_{32}\dot{n} + \xi_{33}T\dot{n} \tag{56}$$

$$t = \xi_{41} + \xi_{42}T + \xi_{43}T\dot{n} \tag{57}$$

$$t = \xi_{51} + \xi_{52}T\dot{n} \tag{58}$$

$$t = \xi_{61} \tag{59}$$

All the growth time models use the scaled T and $\dot{n}$. Model 6 in Equation (59) fits the average of the experimental data and is the model when there is no correlation between T, or $\dot{n}$, and the growth time. If Equation (59) is the most probable, then the current set of models fail to describe a trend in the experimental data. The most probable growth time model, according to Equation (7), will be chosen to find the

Table 35: Results for growth time models after initial experiments

	error	P	ξ_1	ξ_2	ξ_3
(54)	412	0.16	64.8	1.76	–
(55)	365	0.17	62.9	3.63	7.47
(56)	376	0.17	62.0	6.56	–
(57)	391	0.17	66.0	0.52	4.97
(58)	391	0.17	65.9	5.10	–
(59)	414	0.16	64.2	–	–

best next experimental point.

6.1.2 Results of Growth Time Analysis

The results of model fitting for the growth time models after the intial experiments is shown in Table 35. The error for all of the models are all on the same order of magnitude, and none of the models does much better than the others. The initial probability for each of the models was $\frac{1}{6} = 0.166$, showing that the probabilities are relatively unchanged from the experimental data. After including Experiment 7 and 8, the probabilities for each of the models remains unchanged. More experimental data did not clarify which model was more probable, and none of the models fit the data significantly better than (59). Moreover, using the grid algorithm with the initial experiments results in a grid space that is the entire experimental design space.

6.1.3 Discussion of Growth Time Results

This study brings to light to main features of experimental design that must not be ignored. First, the process property you want to model must be robust to noise factors. Second, the methodology has a mechanism for alerting the experimenter when the models proposed are not doing a good job of fitting the experimental data.

From Table 33, one can see that the process growth time was not repeatable. Our initial focus with the CVD testbed was to produce thin films in a short time for the analysis. With the focus on quick deposition time, the robustness of our thin film growth time suffered. One thing that is lacking in this experimental design is a focus on process robustness. This was not a deficiency of the experimental design methodology, but the fault of the objective function that was picked for this process. The objective function for this growth time study is

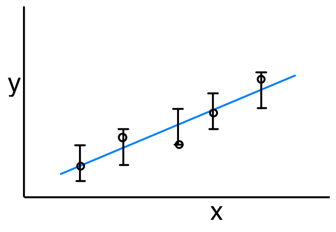
$$\underset{x}{min}(f_{j^*}(x)) = (t(\hat{y}_{j^*}(x)) - t_{goal})^2 \tag{60}$$

one can see that the objective function does not consider the variability of the process. The objective is to solely obtain the desired time, t_{goal} assuming no variability in the process. Including process variability into the objective function could be as simple as adding a term to the objective function

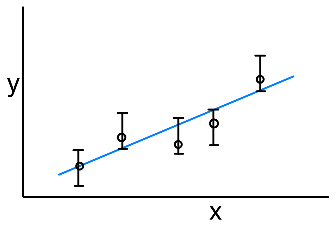
$$\underset{x}{min}(f_{j^*}(x)) = (t(\hat{y}_{j^*}(x)) - t_{goal})^2 + \lambda(\sigma_t(x) - \sigma_{goal})^2 \tag{61}$$

where $\sigma_t(x)$ is the standard deviation of the growth time at an experimental setting x, and σ_{goal} is the desired standard deviation of the growth time. The factor λ is a constant which is adjusted based on the researcher's desired variance. As an additional consequence of making the process more robust, repetitions of every experiment (at least three) would be required to estimate the standard deviation and it would be best to randomize the experiments to make sure process drift does not skew the results. The size of λ can be changed reflecting the importance of robustness relative to performance to the experimenter to better guide the choice of the next experiment. If every experimental point is repeated, one could also use the F-statistic to determine the lack of fit of their model [140].

The variation for every experimental point is needed for the F-statistic to be useful as depicted in Figure 39 where the experimental points are indicated by circles, and the standard deviation on the measurements is shown. In 39(a), the straight line fit to the data is good since the line is drawn through the mean of the standard deviation for each point. This indicates that the data is indeed showing a linear trend. The other case is shown in 39(b). Here, the straight line is still the best fit through the experimental points, but the line does not connect the mean of the standard



(a) Best fit of a line with no lack of fit



(b) Best fit of a line with lack of fit

Figure 39: Two best fit lines. (a) shows a fitted line with no lack of fit while (b) shows a line with lack of fit.

deviations for each point and indicates that a straight line does not actually fit the experimental data very well. Without repetitions for every experimental setting, the lack of fit component of the variation and the variation between experiments cannot be separated.

The case study also brings to light the importance of not only identifying if a model is good, but also identifying if a model is poor. An experimenter can often be certain of their own model's validity without checking it versus other models. Not only does this experimental design include multiple models and their probabilities, but also the grid algorithm can be used as a check of the model's usefulness in the experimental range of interest. If the grid space after the experimental design does not exclude any of the experimental design space area, the model may not be fitting the data very well or else the trend predicted by the model is less than the uncertainty in the model prediction.

Another use of this experimental design methodology is identifying what type of reaction is occurring on the substrate surface. One can operate in the diffusion limited regime or the reaction limited regime of a CVD reactor which were described in Section 6.1. By having a model which is designed for the growth rate in the reaction limited regime and another model designed to predict growth rate in the diffusion limited regime, one can determine which model predicts the growth rate best. The best model will also determine for the researcher which regime they are operating in.

6.2 Analyzing Grain Size with AFM

Here we attempt to analyze the grain size of a thin film using atomic force microscopy (AFM). This has been done numerous times before, so grain size was believed to be a robust measure attainable from AFM [13, 16, 68, 78, 134]. In this analysis we examine Experiment 3 from Chapter 5, using AFM scans taken on different days, and with different AFM tips. What was thought to be a routine analysis turned out to be a very complex problem. When one is designing an experiment, extreme care must be taken that the variability from the measurement device is minimized as well. This is often neglected in literature explanation of scientific results but is critical in a statistical approach or in design.

The AFM images were imported into the Gywddion program and the menu `Data Process>Grains>Mark by Threshold..` was used to mark the grains. `Mark by Threshold` means that a height is chosen by the user. Anything above that height is marked and used in calculating the average grain size in the image. To keep some consistency from one image to the next, the height was chosen to include 30% of the total image area for the grain size analysis. A marked image is shown in Figure 40.

We will examine the variability caused from taking AFM images from the PicoPlus AFM machine described in Chapter 2. One can see qualitatively from Figures 41 and 42 that the grain sizes are going to be different. Both images were taken in non-contact imaging mode. Figure 41 is a representative image from the first day's images, and Figure 42 is representative of the second day's images. The settings used for the AFM are in Table 36. The I and P gain on the AFM are very important, especially if one is doing image analysis. A gain too high will cause the image to be very noisy, and the threshold marking may mark this noise as a grain. This will cause the number of grains to be artificially high, and cause the average grain size to be erroneous as well. The results from the first day's scans are shown in Table 37 while the second day's results are in Table 38. Looking at the average grain size for each

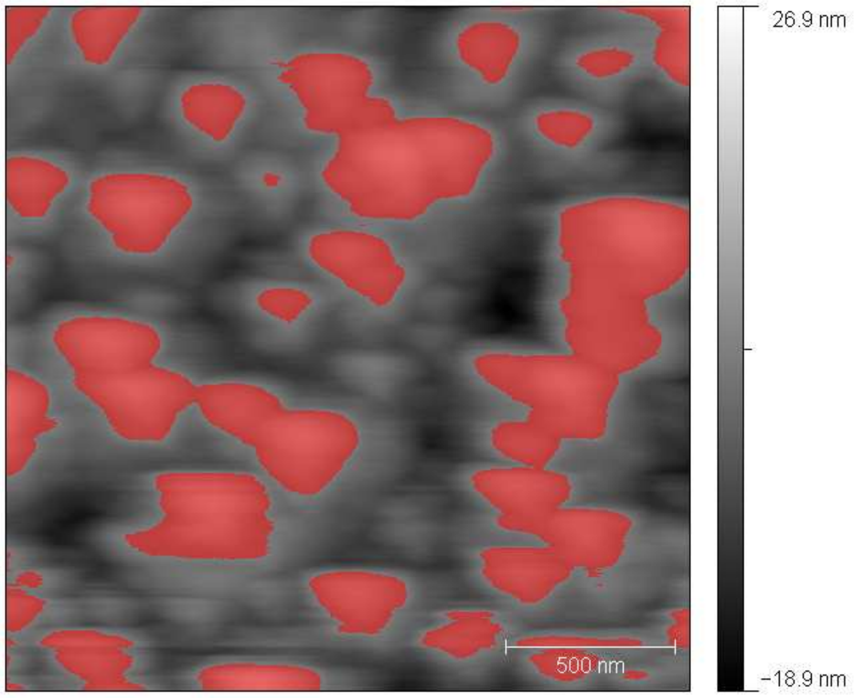


Figure 40: An AFM image that has been marked using the Gwyddion software `Mark by Threshold` function.

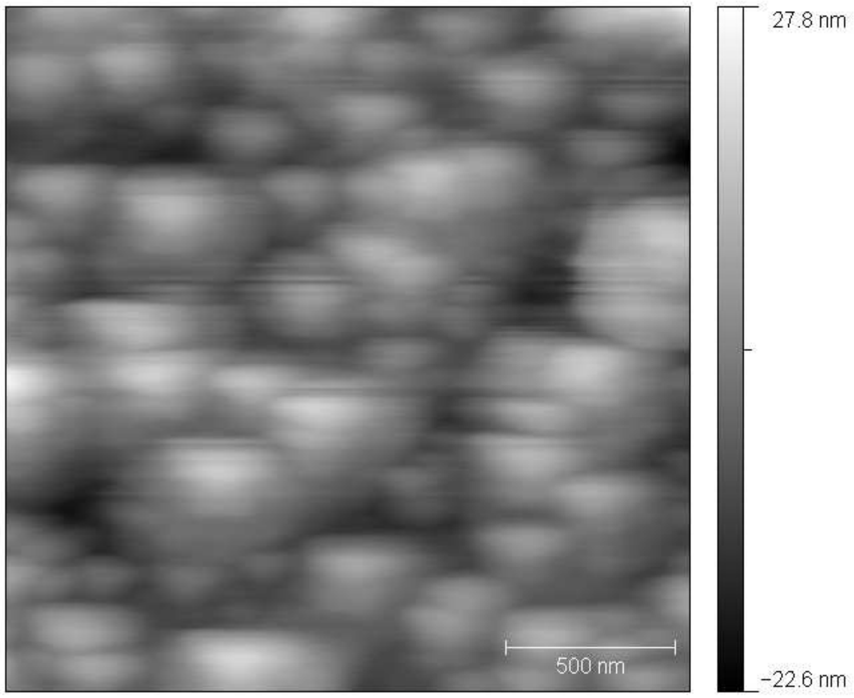


Figure 41: Representative AFM image of Experiment 3 from the first day's imaging, 2×2 μm^2 scan area.

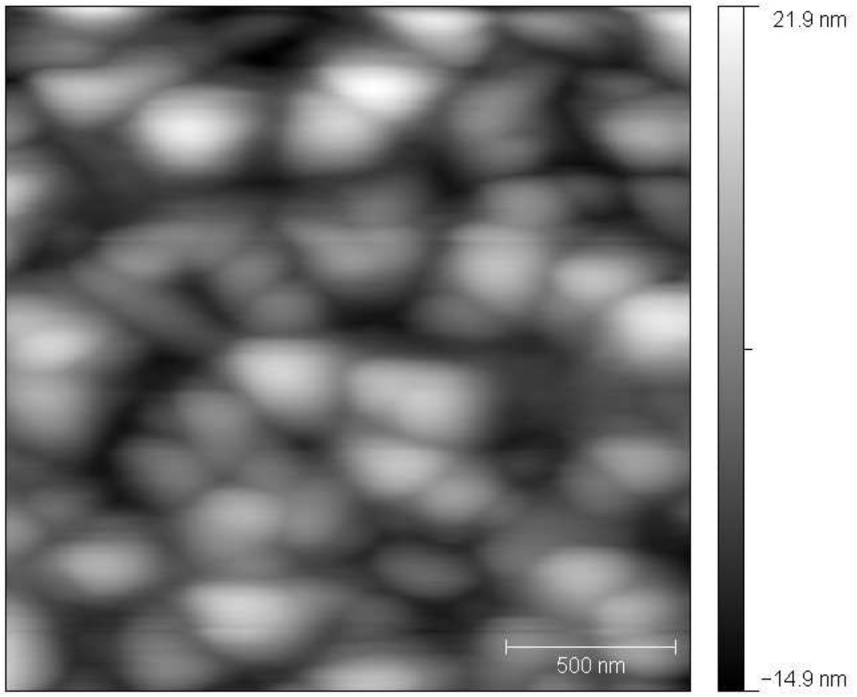


Figure 42: Representative AFM image of Experiment 3 from the second day's imaging, 2×2 μm^2 scan area.

Table 36: Settings used for the atomic force microscope on both days.

Setting	First day	Second day
tip	4	7
resonance frequency	324	325
scan speed $\frac{lines}{s}$	1.02	1.02
I gain	0.1	0.1
P gain	0.05	0.05

Table 37: Results from the first day's AFM for grain size and roughness.

	First day's results					
image numbers	2001	2002	2003	2004	average	standard deviation
grains	50.6	48.1	54.5	59.0	53.1	4.1
% image area	30.1	30.0	30.1	30.9	30.3	0.4
average grain size	113	104	92	95	**101**	8.2
roughness	6.67	6.14	6.74	5.61	**6.3**	0.5

Table 38: Results from the second day's AFM for grain size and roughness.

	Second day's results					
image numbers	3001	3002	3003	3004	average	standard deviation
grains	48	54.3	44.5	46.9	48.4	3.6
% image area	30.2	30.9	30.6	29.8	30.4	0.4
average grain size	145	159	144	117	**141**	15.2
roughness	6.44	6.2	5.84	5.21	**5.9**	0.5

day, the grain sizes vary by 40

Besides the changing tip, there are a number of other possibilities for variation in one day's average grain size to the next. The imaging software may not be the best for detecting grain size. There are many different AFM imaging software available and therefore also many algorithms to find the average grain size. Gwyddion does not specifically outline the grain boundaries of the grains, but rather uses the threshold value in the calculation. Any point that is above this threshold value is considered a grain. Some commercially available software such as the SPIP Grain Analysis module by ImageMet, NIH IMAGE from the NIH website (available for free only as a Macintosh version) [16], Nano Rule + from Pacific Nanotechnology Inc., and other versions available from the various AFM manufacturers. By outlining the grain boundaries in the image, one may be able to attain repeatable average grain sizes from one day to the next.

Other variability from AFM image analysis is inherent to the process. The experience of the user in processing the images plays a role in correcting for artefacts in the AFM image. For instance if a tip is dull, and the tip radius is larger than the feature size one is trying to image, one may actually image the tip rather than the surface.

Another possible path to grain size measurement would be to use the number of grains and calculate the average grain size from this quantity. In Tables 37 and 38, the number of grains from one day to the next is repeatable and does not seem to vary too much from one measurement to the next. The coefficient of variation measures

Table 39: AFM results for Experiment 1

	Experiment 1						
	3001	3002	3003	3004	average	standard deviation	*CV*
grains	35	33	45	25	34.5	8.2	24
roughness	10.5	10.3	9.42	10.9	10.3	0.6	6

Table 40: AFM results for Experiment 2

	Experiment 2						
	3011	3012	3013	3014	average	standard deviation	*CV*
grains	40	42	32	46	40	5.9	15
roughness	6.1	5.0	5.6	5.6	5.6	0.5	9

the variability of the data relative to the magnitude of the average of the data

$$CV = \frac{\sigma}{\mu} \tag{62}$$

where σ is the standard deviation and μ is the average. The CV for the number of grains is 8 in Tables 37 and 38 which is equal to the CV for roughness. All the initial experiments were then analyzed and the data for the remaining experiments can be found in Tables 39-43. The number of grains and roughness are reported along with the CV for each measure. The CV for the number of grains for the other initial experiments are quite high, at one point varying by 25 percent. Since the number of grains is also calculated from the threshold marking by Gwyddion, one has a similar problem as with the average grain size calculation.

From Tables 37–43, one can see that the roughness from one day to the next varies quite a bit less (6%) than the average grain size (40%), and also has a smaller coefficient of variation than the number of grains measurement. For this reason roughness was the metric of choice as there was less variability from day to day

Table 41: AFM results for Experiment 4

	Experiment 4						
	3001	3002	3004	3007	average	standard deviation	*CV*
grains	56	36	38	35	41.3	9.9	24
roughness	7.77	10.6	8.23	8.63	8.8	1.2	14

Table 42: AFM results for Experiment 5

	Experiment 5						
	3001	3002	3003	3004	average	standard deviation	CV
grains	25	19	16	25	21.3	4.5	21
roughness	10.7	11.0	13.9	13.0	12.2	1.5	12

Table 43: AFM results for Experiment 6

	Experiment 6						
	1001	1002	1003	1004	average	standard deviation	CV
grains	24	18	26	17	21.3	4.4	21
roughness	12.4	11.5	11.6	11.2	11.7	0.5	4

characterization using this metric.

6.3 Conclusions

In this chapter we have addressed a few important concerns in experimental design. When using an experimental design method which makes use of a model, it is important to realize that "all models are wrong, some models are useful" [19], meaning one should not rely too much on a model. To ensure that one does not put too much confidence into the model, a model which outputs the average of the experimental data was included among the viable models. The usefulness of this model checking was shown in the growth time modeling. This section also highlighted the need for robustness to be included into experimental design to make sure results are useful for engineering a process. In the last section, we highlighted the importance of choosing a good metric upon which to base the model and experimental design. In trying to construct a model to predict grain size, the variance of the data from one day to the next was too great, and a more robust metric (roughness) was chosen for the model.

CHAPTER VII

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

When this project began, there existed a chemical vapor deposition system that was completely uncharacterized, and a desire to construct a microstructural model based on processing conditions. The first step was to study the existing microstructural models to see what had been and what would be useful for this project. In searching through the existing models, there were two types of models present. The first model type was purely empirical. After many experiments some conclusions were drawn about the processing conditions of a process and parameters were fitted to the experimental data. The parameters were sometimes assigned some physical meaning and were sometimes explained using some insight into what occured in the process. These models captured what was happening in the experiments that had been run, but were not useful for explaining how the process works and were not appropriate for extrapolation.

The other type of model found was mechanistic in nature and very computationally intensive. These models were grounded in theory and derived from first principles. Simulations could be run with these models that seemed to match what was actually going on in experiments. The drawback of these models was the disconnect from experimental data. The processing parameters for these models had no physical meaning such as flux of atoms to a substrate or were immeasurable with the current technology and technology typically found on equipment such as grain boundary energies. Furthermore, the details of the models had to be understood well to transfer it to another system or to use the model with a physical system.

The overall purpose, then, was to bridge these two types of models. The empirical models are useful for explaining what happens in experiments, but do not provide insight into what is happening on the molecular level. The mechanistic models are useful for explaining what happens at the molecular level, but does not always relate well to actual experiments, especially when there is limited data available. What if these two types of models could be blended? A model could be constructed which was useful for explaining experimental data and used physical process parameters, yet was also capable of predicting what was occuring on the molecular level and was useful for extrapolating outside of the current experimental data.

The next question to answer was how to go about constructing such a model on a system when there is very little prior knowledge. To this end experimental design was explored. By planning the experiments well to get the most information out of them, one could hopefully construct such a model quicker. In studying the types of experimental design, another dichotomy similar to the model types was apparent. There were experimental designs to construct empirical relationships between the output and the processing conditions. The other experimental design type focuses on the model for the process and more accurately determining the parameters of the model. For this design to function, one had to have a process model which one was confident in to direct the next experiments. But how does one design experiments to construct such a process model when there are none available?

This work fills in the gaps between empirical and mechanistic models and experimental design and is unique in that the work was guided by the experiments. The experiments kept the experimental design simple. The procedure to run the experimental design methodology is basically the same from one study to the next. One needs to provide a functional form for the different models to be included in the procedure (number of parameters, initial guess for parameters, range of possible values for parameters, etc), but this is work the researcher will be doing anyway if they are intersted in developing a process model.

The experimental component also guided us to develop a method that could be used in practice. What were people doing in practice and how can we improve on that? The methodology is versatile since there is not just one experimental design that is best for all systems. Different researchers have different experimental design preferences and different objectives for their experiments, so this methodology had to be adaptable for many different situations, not just useful for the CVD process used to test the theories. The result of this work is a unique view to experimental design that bridges the empircal and model-based methods of experimental design. The methodology developed is a tool to be used in constructing process models that relate to the physical process while also providing insight into the phenomena occuring in the process.

In Chapter 3 current experimental designs are explored and a new methodology is developed. The gap between empirical and mechanistic experimental design methods is explored and new methods are developed to find the optimal operating point of a process. A model discrimination method was developed but it designed experiments using the worst model and the best model of a given set of models. The experiments are not necessarily near the optimal point of the process and do not provide information about the optimal point. If two models disagree at a given point, this may help in discriminating between models, but does not necessarily help to find the optimal point of the process. For example, the two models may agree at the optimal point, making this model discrimination method undesirable for this work. For this reason, model discrimination was not included in further methodology development.

A revised methodology is developed to focus experiments near the predicted optimal point of a process. The grid algorithm is developed which makes use of the confidence interval around the optimal point to reduce the experimental design space. The objective function for this experimental design will change depending on the goal of the experiments, but for this work the objective is to produce a system output close to the target value for the process. The methodology also incorporates multiple models as a way of checking the usefulness of the model. If your best model is still bad, the methodology indicates one is in need of modifying the existing model, finding a new model, or need for running more experiments. By incorporating traditional design of experiments used by experimentalists, the method is straightforward and easily adaptable to different experimental situations.

In Chapter 4 the revised methodology is shown to function on simulated data on a modified Himmelblau function and how it functions on a nucleation model for thin films. In both simulation studies, three different experimental designs are compared: D-optimal, P-optimal which minimizes the prediction variance about the predicted optimal point, and a random experimental design. The three designs are used with and without the grid algorithm. In both studies, the random experimental design plus the grid algorithm performs best for finding the optimal point of the process. Some parameters for the grid algorithm are also explored in this section such as the grid size, the desired level of confidence on the stopping criteria, and the effect of the initial set of experiments. It is emphasized that this methodology does not replace the good sense of the experimenter, but is merely a tool to help the experimenter meet his goals. Depending on those research goals, one can choose different parameters for the methodology. The methodology was also compared with a "greedy" experimental design objective where instead of sampling near the optimal point, the greedy method actually samples at the optimal point continually to make the model fit better to the optimal point. Here, the grid algorithm performs similarly as the greedy method, but has the advantage of exploring more of the experimental region. If the model is incorrect, the greedy method would not explore other experimental parameters if the predicted optimal point does not change and therefore may not identify if a model performs poorly.

In Chapter 5, this methodology is implemented on an actual chemical vapor deposition system. The purpose is to find the optimal point of the process while building a useful process model. Six initial experiments are performed and two experiments are designed using the sequential methodology, while models for roughness are used to find the optimal point of the process. Hybrid models are developed relating the nucleation density to the resulting film roughness. The best fit model is an empirical model relating T to the resulting film thickness which agrees with the analysis of the intial experimental data findings. A hybrid model is also a probable fit to the experimental data indicating that roughness is indeed related to the nucleation density. The sequential experiments reduce the confidence interval around the predicted optimal point.

In Chapter 6 various aspects of experimental design are explored. The need for model checking within the experimental design is highlighted by an attempt to fit growth time of thin films grown in the experimental testbed. A statistical F-test could also fix this problem, but the test requires experimental repetitions to be effective, which a researcher does not always have the luxury to do. Many experimental design methods assume that the model used for the design is correct, which is not necessarily the case when one is constructing a new process model with mechanistic components. Here, the method did not fail, but large confidence intervals and a large grid region indicated uncertainty on the optimum from the equipment, not just from the models being used. Also, the need for a robust metric upon which to build a model is explored. Here, the grain size of the thin films measured by AFM is used as a case study. The analysis showed that grain size via AFM shows potential, but is not robust in its current form.

This work has unique contributions to the field of model building, process design, and experimental design. This is the first experimental design methodology to our knowledge with the specific aim of developing process models. Mechanistic models are often developed removed from experimentation, and the result is a model that does not relate well to the actual process and cannot be applied to the process directly. Alternatively, experimental data is used to develop empirical models, but these models are only useful in the range where the data was collected and should not be used to extrapolate into other regions. By anchoring the model in experiments while also testing hypotheses for deeper understanding of the process, a practical model for experimenters can be developed.

This work also demonstrates the need for researchers with a diverse skill set. While many are either experimentalists or theorists, there is a definite need for researchers capable of working in both realms or at least capable of working closely with another researcher with a complementary skill set. Models developed without experiments will be unlikely to have a practical application, while models developed without theory will have limited predictive range.

7.2 Future Work

Due to the dual nature of this project-having an experimental and a theoretical component-there are two general directions one could go in future work. These two directions are not necessarily mutually exclusive, but working in a group would be very beneficial. One part of the group could focus primarily on experimental work, while the other part of the group could focus on furthering the theoretical work. For this reason, the two directions with the associated possible projects will be described in different sections.

7.2.1 Experimental Work

The CVD testbed shows promise for producing a wealth of experimental data for experimental design and for *in situ* sensing, but still needs improvement on reproducibility. After a robustness study which was described in an earlier chapter is run and consistent growth rates are obtained, it would be interesting to explore dynamic settings on the system and their effect on growth rate. Currently, the temperature, pressure, and gas flowrates remain constant throughout the deposition process. By changing these settings during a deposition, one can gain greater insight into what affects the growth rate of the film, and also observe the changes in microstructure. This greater insight could lead to more descriptive models of the deposition process, especially for the nucleation phase. Most of the current models for thin film nucleation involve parameters that are not measurable during a routine deposition such as flux of atoms to the surface, how many nucleation sites are present on the surface (this may be determined for *ex situ* analysis, but would be difficult to find *in situ*), and activation energies of the various surface processes. A more descriptive model of the deposition process could open the door to further process optimization.

The *in situ* sensor was quite underutilized in this research. The capabilities of the emissivity-correcting pyrometer (ECP) were demonstrated in [149] for state estimation. A controller developed using this model would enhance further studies of thin film deposition and improve the repeatability of the process. The ECP was very useful in this study to estimate the thin film thickness and further shows the need for *in situ* sensor development for CVD processes. The drawback to the ECP is the need for a nearly perpendicular line of sight to the substrate. This requirement prevents the use of a typical showerhead in the CVD reactor and showerheads have been shown to greatly enhance the film uniformity [131].

The reactor system is capable of depositing other thin films as well. The system is equipped with two evaporators and two ovens, making depositing other thin films, especially ones where the precursors sublimate at different temperatures, possible. Other thin films such as In_2O_3 or GaN are useful for microelectronic applications, and yttria-stabilized zirconia (YSZ) is still a possibility and was the thin film in mind when the reactor was originally designed. The interesting aspect of YSZ is Y composition control and the effect on the resulting microstructure. The composition can be monitored via x-ray photoelectron spectroscopy (XPS) and the processing conditions on the resulting composition can be explored. XPS is an ultra high vacuum technique that measures the electrons emitted from a thin film after exposing it to x-ray. The kinetic energy of the electrons are measured and used to determine the composition of the top 10 nm of a thin film. XPS can also be used to obtain compositions at different depths of the thin film, but the sputtering process used to reach those depths is destructive to the thin film.

The crystallographic structure of the thin films was largely unexplored, but could be obtained through x-ray diffraction (XRD). XRD uses x-rays in a non-destructive manner to find the crystal structure of solids. An x-ray beam is incident on the surface at an angle θ and the diffraction of the beam is collected. Using Bragg's law $n\lambda = 2d \times sin\theta$, the distance d between the different planes is calculated. This is a very useful method for identifying materials and determining the properties of the thin film. For example the grain size can be estimated using the Scherrer method. The crystal structure of a thin film also affects the properties of a thin film, making it desirable to include these properties in a model for the microstructure of a thin film. For this work there was too much uncertainty in identification of the microstructure by XRD. However, given more time and more guidance and expertise in XRD these properties could be included in a microstructural model.

Another analytical technique that should be further explored is atomic force microscopy (AFM). In this work it is shown that the method for acquiring grain sizes from AFM data can be tedious and very user dependent. However, with software capable of finding the grain boundaries on the surface of the thin film, better estimates of the grain size can be obtained. The grain size is of particular interest for the application in thermal barrier coatings where small grain sizes decrease the thermal conductivity of a thin film [119]. Developing a model which can predict grain size would be beneficial, especially if the analysis can be performed with an AFM which is generally a cheaper analytical technique per hour than other methods such as scanning electron microscopy (SEM) and XRD. AFM is also non-destructive to the sample, does not require further processing of the substrate for analysis, and is very easy to set up.

Experimentally, the experimental design method designed in this work can be applied on any physical batch process. While it was developed with microelectronics in mind, many other areas could also benefit from the experimental design. In particular bioprocesses which were mentioned in the introduction could benefit from this work where the exact reaction mechanism is unknown. Here a part of the reaction is modeled using neural networks, but could benefit from some more knowledge in how the reaction actually works. By testing multiple hypotheses at once and observing the performance of the different models, more insight into the system could be acquired.

7.2.2 Theoretical Work

The simulation studies in Chapter 4 were the first attempts to study this methodology, but many possibilities for simulations still exist. The current MHF simulation used a model to fit the experimental data where the model form predicted a similar global optimum as the real system. By changing the model, one could obtain a model that does not predict the same global optimum as the system. How would the experimental design perform in this case? Would the methodology provide insight to the researcher on how to correct their model?

Other simulation studies with the same systems as Chapter 4 would also be possible. One could study the effect of repetition on the experimental design performance, particularly in trying to include process robustness as well as an optimal point for the process. The effect of the initial guess for the model on the experimental design was also untested. If the researcher has a bias towards an incorrect parameter guess, would that affect the performance of the methodology?

The experimental design methodology developed here is also not finished developing and there are many possibilities for the future of experimental design. In this work the development of experimental design was influenced heavily by experimental work being performed simulataneously. In doing this a shift in the way one looks at experimental design has occurred and the unique perspective from performing experiments should not be neglected in further experimental design research.

The objective function for the experimental design can also be modified. In this work, the objective function was constructed to find the optimal point of the process, but other objectives could also be incorporated. Model discrimination was not used in this methodology, but that does not mean that it could not be incorporated into the objective function as well. The model discrimination attempted here compared models pairwise, but one could also incorporate all of the models at once [24]. As mentioned in Chapter 6, the objective function could also be modified for process robustness. The objective function is project specific and can be changed to fit the particular needs of the experimenter. A good objective function is also essential to successful experimental design, otherwise one is designing experiments which they do not actually need!

An enhancement of the experimental design could be accomplished using measurement uncertainty. In the current work, the uncertainty of the measurement technique was not included or propagated to the final confidence interval of the model. Realistically, this measurement uncertainty should be included for more accurate confidence intervals on the models. This could also draw attention to the need for better measurements and identify which characteristics of a thin film or any material needs to be measured with better precision.

Bayesian experimental design was not included in this work, but could be a valuable addition to the methodology. Bayesian experimental design is particularly helpful when the uncertainty of the model is well defined and informative prior probabilities can be calculated. Unfortunately, the big hindrance of wide Bayesian experimental design lies in the mathematically intensive development of the prior probabilities which is not straightforward. If a more simple method of calculating informative priors could be found, use of Bayesian methodology would definitely grow.

The stopping criteria for the experimental design presented in this work is simple, and improvements in deciding when to finish experiments are needed. There is a surprising lack of research on the question of when is a model good enough or when to decide to abandon the model altogether. For this reason there are a multitude of questions one could answer in future experiments.

- One can continue running experiments to reduce the confidence interval, but is the added information profitable given the objective of the experiments?
- Another operating point may be a potential optimal point, but is the current operating point just as good?
- What is the risk of not doing more experiments? The expected information from an experiment was quantified in [15] given an appropriate utility function for the experiments.
- The probability of another experiment changing the end decision of the experimenter is the quantity of interest.
  - How sure is the researcher that his hypothesis is correct?
  - Would he change his conclusion if another experiment was executed that refuted his hypothesis, or would the new information be discarded as erroneous?
  - How much information would the next experiment have to provide for the final conclusions to change?

APPENDIX A

RCA CLEANING PROCEDURE

1. Mix solutions

   (a) Organic clean solution DI H_2O:NH_4OH:H_2O_2 (5:1:1) SAFETY: always add acids/bases to water

      i. 100mL DI H_2O

      ii. 20mL NH_4OH

      iii. 20mL H_2O_2

      iv. Solution is put on a hot plate to 80^{o}C

   (b) Oxide strip solution DI H_2O:HF (50:1) SAFETY: HF is very dangerous. Use neoprene gloves when handling HF container. If HF is spilled, contact emergency personel and take person to hospital. Flush region with water and use HF spill kit to clean affected area.

      i. 150mL DI H_2O

      ii. 3mL HF

   (c) Ionic clean solution DI H_2O:HCl:H_2O_2 (5:1:1)

      i. 100mL DI H_2O

      ii. 20mL HCl

      iii. 20mL H_2O_2

      iv. Solution is put on a hot plate to 80^{o}C

   (d) DI H_2O rinse containers Fill two beakers with 150mL DI H_2O

2. Procedure

(a) Once solution is 80^{o}C, put wafer in wafer dipper and place in organic solution for 10 minutes

(b) Take wafer out of organic and place in 1st H_2O rinse beaker for 5 minutes.

(c) Place wafer in oxide strip solution for 15 seconds.

(d) Place wafer in 2nd H_2O rinse beaker for 1 minute.

(e) Place wafer in ionic clean solution for 10 minutes.

(f) Place wafer in 2nd H_2O rinse beaker for 10 minutes.

(g) Blow dry wafer with N_2 gas.

APPENDIX B

CVD PROCEDURE

1. Sign in to logbook!
2. OPEN Labview
3. TURN ON pump
4. OPEN Ar, O2 and N2 containers
5. RAISE the pressure in the reactor by flowing Dummy Ar into the reactor (Use 3000-4000 sccm)
6. PUT ON gloves
7. REMOVE heater from the reactor
8. PLACE previously grown film into a case, and date it
9. CLEAN new wafer
10. Follow RCA procedure on fume hood (see Appendix A
11. RINSE wafer with DI water
12. DRY wafer with N2
13. Remember to CLEAN and DISPOSE of waste properly
14. SAVE file
15. CLICK the Comm On and LEDs On under the Engine Tab in Labview

16. CLICK Calibrate Low under the Reflectance Tab

17. LOAD new wafer on the heater and PLACE it in the reactor

18. REMOVE gloves (optional)

19. TURN ON voltage device

20. REATTACH thermocouple and voltage wires

21. TURN ON the TV

22. PUMP down the reactor to ¡ 1 torr

23. ADJUST the camera so the spots appear in the center of the TV screen.

24. FLOW Dummy Ar at specified amount (Default: 100 sccm)

25. CALIBRATE high reflectance by clicking that button in the Reflectance Tab

    (a) Put R470 at 0.45261

    (b) Put R 950 at 0.3371

    (c) Write down both Intensity Lows and both Intensity Highs in case computer crashes

26. CALIBRATE UV Cell

    (a) TURN ON UV Lamp. Flip switch on UV cell panel

    (b) TURN ON Master, Slave 1, and Slave 2 under the Ocean Optics Tab

    (c) Change integration time to maximize 280-300 range (about 160)

    (d) TAKE out UV cord from the source

    (e) COVER the cord end with your finger and click Store Dark

    (f) SCREW cord back

(g) STORE Reference

27. TURN ON main oven (Default: 400 F) MAKE COMMENT IN LABVIEW WITH SETTING!

28. TURN ON fan (level 1) and blow it on the valve (side with N2 Trap)

29. SET voltage and ramp rate (Default: 15 V at 0.35 V/min)

    (a) Take voltage halfway to desired voltage. Only bring heater up to desired voltage when youre ready to run experiment

30. WAIT for system to heat up (Time: 90 min)

    (a) Dispose of chemicals in hood in proper waste containers

    (b) Cover HF solution with parafilm to use later.

31. After 60 minutes

    (a) SET actuator temperature, the green numbers next to the low pressure meter (Default: 150-200 C) MAKE COMMENT IN LABVIEW WITH SETTING!

    (b) Open O2 and N2 cylinders

    (c) Once evaporator temperature is 30 degrees below desired T, reduce oven T from 400 to desired level

32. FILL N2 trap 30 minutes before opening evaporator

    (a) WEAR big blue gloves

    (b) PLACE tank tube into the liquid N2 container

    (c) SLOWLY open liquid N2 tank, then once gas has passed and it cools down open it more

   (d) SLOWLY pour liquid N2 into the catcher on the left side of the oven

   (e) MAKE COMMENT IN LABVIEW WITH SETTING!

33. FLOW O2 (Default: 50 sccm)

34. FLOW Carrier Ar (Default 20 sccm)

35. PUT ON white heat resistant gloves

36. OPEN precursor chamber exit valve

37. OPEN valve leading into the precursor chamber MAKE COMMENT IN LABVIEW!

38. Calibrate Ratio Temperature by clicking Calibrate Ratio Temp under Temperature Tab

39. FLOW Precursor Ar (Default: 20 sccm)

40. CHECK absorbance in the Ocean Optics Tab to ensure precursor is flowing

41. *If necessary, ADJUST Ar flow rates and valve openings to achieve desired absorbance

42. SWITCH solenoid valve to Yes to start deposition (make sure N2 tank is open)

43. WAIT as film grows

   (a) WATCH absorbance to make sure precursor doesnt run out

   (b) WATCH Ar and O2 flow rates to make sure they are present

44. MONITOR Labview for any problems

    Steps 45 and 46 should be carried out concurrently

45. SWITCH solenoid valve to No to stop precursor flow to the reactor

(a) TURN OFF Precursor Ar flow

(b) PUT ON white heat resistant gloves

(c) CLOSE valve entering the precursor chamber

(d) CLOSE valve exiting the precursor chamber

(e) MAKE COMMENT IN LABVIEW!

(f) WAIT for Absorbance to go down to zero then TURN OFF Carrier Ar, may need to pulse carrier Ar to remove precursor from line

(g) TURN OFF the main oven MAKE COMMENT IN LABVIEW!

(h) SET the actuator temperature to 25 C MAKE COMMENT IN LABVIEW!

(i) DISABLE the master and slaves under the Ocean Optics Tab

(j) TURN OFF the UV Lamp

46. WAIT for reflectance to stabilize indicating the end of deposition

    (a) TURN OFF O2

    (b) STEP the voltage to zero

47. Wait for voltage to reach zero

    (a) DISABLE Comm and LEDs under the Engine Tab

    (b) TURN OFF the TV

    (c) TURN OFF the voltage source

48. OPEN the ovens and MOVE the fan to blow into oven (Fastest fan spd. is 3)

49. WAIT until heater temp ¡200 C (Time: 20-30 min)

50. TURN OFF Dummy Ar

51. CLOSE Ar, O2 and N2 tanks

52. CLOSE Pump Valve on Labview

53. TURN OFF the Pump

54. QUIT Labview

55. TURN OFF the Fan

56. GO HOME. You deserve it!!

APPENDIX C

FUTURE REACTOR OPERATION NOTES

If experiments were to continue on the CVD reactor, here are a few thoughts of how to better operate it in the future.

1. General Notes

   (a) Journal articles do not provide specific details about reactor operation. In the future more theses should be explored for more information about operating a CVD reactor. Theses are more likely to include failures as well as successes which are very useful for one learning how to operate a new piece of equipment.

   (b) Run proof of reactor experiment before trying to run experiments. If possible, find a material to deposit other than yttria or zirconia since the precursors are very expensive.

   (c) When learning a new piece of equipment, start out simple. Too much time was wasted trying to deposit yttria-stabilized zirconia when a proof of reactor experiment had not yet been performed.

   (d) A robustness study on deposition time would have been helpful to allow modeling of deposition time.

2. Evaporator specific suggestions

   (a) Load evaporator with 10 g instead of 0.5 g to allow for longer time between changing evaporator. Some variability may have been reduced had the evaporator not been replaced so often. Longer use of evaporator would also allow for more rapid experimentation as loading an evaporator was a half-day process.

(b) With longer evaporator use each time, would allow for more regular cleaning of evaporator with procedure outlined in [131].

(c) Plan experiments to not allow precursor to decompose in evaporator. Excessive time in between experiments (due to their being only one operator) could also explain some of the variability in reactor operation from one day to the next.

3. Reactor operation notes

   (a) Try to recreate diffusion vs. reaction-limited curve for this reactor for positive identification of which region the reactor is operating in (citations).

   (b) Try reducing argon flow through the evaporator. The molar flow rates set in this thesis are higher than the setpoints in other research (citations).

   (c) The growth rates seem to go down as total argon flow was increased. Try also reducing total argon flow.

APPENDIX D

MATLAB PROGRAM

D.1 Matlab code for MHF simulation study in Matlab

This is the file used for the MHF simulation study.

```
function ni=trial23(err,flag2,rep,sctol,step1,num)
%%%%%%%%%%%%%%%%%%%%%%%%%%%
%  This is a simulation to see whether D, G optimal, or factorial
%  design does best when looking at prediction variance, parameter
%  estimation/certainty/accuracy, accuracy of optimal point, goodness of
%  fit/ability to predict future observations
%  This simulation has a different form for the models (i.e. the # parameters
%  is the same, but where they are in the model has changed.  Trying to avoid
%  inv(J'J) being rank deficient
%%%%%%%%%%%%%%INPUTS
%matflag= flag for Daugment matrix (depends on mod)-matflag not used at
    %the moment    (9/24/07)
%num=number of trial for saving purposes
%err=variance of noise added
%defines tolerance for stopping criterion
%flag2= vector of length 4 for each optimality, tells whether or not to run box algorithm [P D G
   Run] 'Run' is a flag to say whether to run stopping criteria or not, 1=yes.  Included with SC is
    whether to run optimal point experiment or not.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%List of variables
%a, auggie1, auggie2,b,c,cells1, counter, d, D, data, first, flag, ,
%goal,i, init, iterd, iterg, iterp,j, jp, J, J1, Low1,LB,m,min, max,
%mod, n, nD, ni, num, opt, optrange, out, output, P, po, point,point1,
%pointopt, pointp,PV,
%range, second, source, step, step2
%Srhat, sum, thetahat, time, Up1, UB,wkdir,xnew, xd, xo,y,yd,yp,yg,z

%mod is the name of the model function we are using, should be in form '@mod1'
%if 'mod1' is the function name for the model.
%% experimental settings
source='trial23';          %Keeping track of what generator file generated what data
xmin=-5; %lower bounds for experimental settings x1 and x2
xmax=5;  %upper bounds for experimental settings x1 and x2
optrange=[-5 5];              %range for experimental settings for graphing function
step2=30;                     %finding optimal point, needed more spots on grid
step3=15;                     %for p-optimal grid
ni=16;               %max # of iterations
cells=[17 19];         %defines size of cell arrays for opt and model arrays
po=[-1 -1 -1];              %initial guess of parameters for newfit.m
LB=-70*ones(1,3);           %range for parameters
UB=250*ones(1,3);
xo=[0 0];                   %initial guess for next experiment
pointopt=[-3.8 -3.32];  %coordinates of optimal point in MHF
```

```
opt=43.3;                  %value at optimal point
mM=5;                      %number of models in simulation
output=198.7;              %this is the starting avg. output of system
time=1;               %counter for keeping track of optimal outputs
goal=sctol(1);
rand('state',sum(100*clock));             %need to initialize seed for random number generator
%cell arrays for data storage are dopt, ropt, gopt, popt
%preallocating space for each opt array
dopt=feval(@cell,cells(1),1);
ropt=feval(@cell,cells(1),1);
gopt=feval(@cell,cells(1),1);
popt=feval(@cell,cells(1),1);
init=feval(@cell,cells(1),1);      %extra cell array for inital calcs
%preallocating memory for the cell array for models
model=cell(cells(2),mM);
modelr=cell(cells(2),mM);
modeld=cell(cells(2),mM);
modelg=cell(cells(2),mM);
modelp=cell(cells(2),mM);

% Define model cell array for each model
model{1,1}=@mod7;
model{1,2}=@mod6;
model{1,3}=@mod9;
model{1,4}=@mod10;
model{1,5}=@mod11;
model{14,1}=3;
model{14,2}=3;
model{14,3}=3;
model{14,4}=3;
model{14,5}=3;
model{2,1}=po; model{2,2}=po; model{2,3}=po; model{2,4}=po; model{2,5}=po;
point=zeros(5,2);
dout=cell(1); gout=cell(1);  pout=cell(1);       %initializing pout, dout, and gout
allP=zeros(mM,1);
Pvalue=zeros(ni,1);
iprob=1/mM;                     %initial probability for all models so they're
                                %equally probable

%opt cell arrays store info as follows:
%{1}=d                          max value of experimental data
%{2}=data                      stores the simulated data
%{3}=err                        stores error
%{4}= range                     range for values of parameters
%{5}= optrange                  range for experimental points (upper and lower
%bounds)
%{6}=experimental point         next experimental point to be run
%{7}= # repetitions to be run
%{8}= element to pass value of 'a' to other functions
%{9} stores number of correct model for each iteration
%{10} stores max value acceptable for box calc
%{11}=box xrange                  range for box calc
%{12}=box yrange                  range for box calc
%{13}=system output for stopping criterion
%{14}=verdict                      tells user why simulation was stopped
```

```
                              %1=theta and Srhat not changing
                              %2=CI < sqrt(noise)
                              %3=CI < CI (desired)
%{15}=counter                      counter for stopping criterion, see SC for all
                                   %stopping criterion questions
%{16}=point outputted by box1.m function
%{17}=tolerance for SC

%cell arrays for each model
%1=model handle
%2=thetahat
%3=srhat
%4=normalized probability of model
%5=estimated optimal point
%{6}=J                             Jacobian of data with respect to parameters
%{7}=D                             D-optimal value of model
%{8}=PVxnew                        prediction variance at new experimental point
%{9}=PVxopt                        prediction variance at estimated optimum point
%{10}=PVopt                        PV of real optimal point (for comparison)
%{11}=errorxopt                    model error in predicting optimal point
    %f(x1,x2)
%{12}=model prediction at estimated optimal point
%{13}=model prediction at true optimal point
%{14}=# parameters in the model
%{15}=distance from true optimal point
%{16} holds F statistics for LOF
%{17}=experimental error (calculated)
%{18}=confidence interval on predictions
%{19}=studentized residuals
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%% start with 5 initial points
        data1=MHF([0 0],rep+1,err);
        data2=MHF([xmax xmin],rep,err);
        data3=MHF([xmin xmax],rep,err);
        data4=MHF([xmax xmax],rep,err);
        data5=MHF([xmin xmin],rep,err);
        data=[data2;data3;data4;data1;data5];
        [n,m]=size(data);
        d=max(round(data(:,3)));                %calculates maximum of output for jacobian
          calculation
        %preset the opt cell arrays to the correct initial values
        init{2,1}=data; init{1,1}=d;  init{3,1}=err; init{4,1}=[LB; UB]; init{5,1}=optrange; init
          {11,1}=optrange; init{12,1}=optrange; ropt{13,1}=output; ropt{15,1}=0; init{7,1}=rep;
          init{6,1}=[xmax xmax];
        init{8,1}=1; init{17,1}=sctol(2);        %set init 'a' to 1 for initial calculations
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        %Parameter fit
        for i=1:mM
                [model{2,i}(1,:),model{3,i}(1),model{5,i}(1,:),model{6,i},model{7,i}(1),model{8,i
                  }(1),model{9,i}(1),model{10,i}(1),model{11,i}(1),model{12,i}(1),model{15,i}(1),
                  model{13,i}(1),model{16,i}(1,:),model{17,i}(1),model{18,i}(1),resid{i,1},init
                  {1,1}(1)]=metrix3({model{:,i}},{init{:,1}},step2);  %This calculates all
                  quantities needed to run the box algorithm
                %calculate initial probabilities of each model
                [allP(i)]=mprob(iprob,{model{:,i}}, {init{:,1}});
        end
        %calculate initial Bayesian probability for each model after
        %initial experiments and determine which model is most probable
        [P,init{9,1}(1)]=maxP(allP);
        %store initially correct number in each optimality array
            popt=init;   ropt=init;        gopt=init;        dopt=init;
        for i=1:mM
            model{4,i}(1)=P(i);
        end
        %assign initial values to each model array for each optimality
        modelr=model; modeld=model;modelg=model;modelp=model;

%% For loop for random experiments
out=0;            %preset 'out' to zero
count=0;          %preset to zero
for a=1:ni
    ropt{8,1}=a;          %stores a value for use in other functions
    iterr=a;            %for graphs and output.
        if flag2(2)==1
                [pointr,XY4,Z4]=optgraph(ropt{5,1}, ropt{5,1}, step1,modelr{2,ropt{9,1}(a)}(a,:),
                  {modelr{:,ropt{9,1}(a)}},{ropt{:,1}},modelr{17,ropt{9,1}(a)}(a), 3);
                [rout{a},ropt{16,1}(a,:),ropt{10,1}(a)]=box1({XY4{:}},Z4,{ropt{:,1}},{modelr{:,
                  ropt{9,1}(a)}},3);
                [o,q]=size(rout{a});
                if o>1
                        pop=round(unifrnd(1,o));
                        ropt{6,1}(a,:)=rout{a}(pop,3:4);
                else                        %if points can't be found on the grid, then make random
                  points
                        ropt{6,1}(a,1) = unifrnd(optrange(1),optrange(2));
                        ropt{6,1}(a,2)= unifrnd(optrange(1),optrange(2));
                end
        else
                %generate experimental points using random number generator, unifrnd picks random
                  number between optrange
                ropt{6,1}(a,1) = unifrnd(optrange(1),optrange(2));
                ropt{6,1}(a,2)= unifrnd(optrange(1),optrange(2));
        end
        %generate new data point
        newdat=MHF(ropt{6,1}(a,:),rep,ropt{3,1});
        %add to existing dataset
        ropt{2,1}=[ropt{2,1};newdat];
        for i=1:mM
                [modelr{2,i}(a+1,:),modelr{3,i}(a+1),modelr{5,i}(a+1,:),modelr{6,i},modelr{7,i}(a
                  +1),modelr{8,i}(a+1),modelr{9,i}(a+1),modelr{10,i}(a+1),modelr{11,i}(a+1),modelr
                  {12,i}(a+1),modelr{15,i}(a+1),modelr{13,i}(a+1),modelr{16,i}(a+1,:),modelr{17,i
                  }(a+1),modelr{18,i}(a+1),residR{i,a},ropt{1,1}(a+1)]=metrix3({modelr{:,i}},{ropt
                  {:,1}},step2);
                allP(i)=mprob(iprob,{modelr{:,i}}, {ropt{:,1}});
        end
        %calculate normalized probabilities for each model and find most
        %probable model
        [P,ropt{9,1}(a+1)]=maxP(allP);
        for i=1:mM
                modelr{4,i}(a+1)=P(i);
        end
        exper=size(ropt{2,1});       %calculates how many experiments have been run
        [out,count,ropt{14,1}(a,:)]=SC2({modelr{:,ropt{9,1}(a+1)}},ropt{8},goal,count,exper(1),rep
           ,ropt{17});
        if out==2
                if flag2(4)==1
                        break
                else
                        out=0;            %if SC is not being run, change out to zero so box
                           algorithm can be run
                end
        end
end
clear point
OPTIONS = optimset('DiffMinChange',1e-6,'Display', 'iter','Diagnostics', 'ON');
%% For loop for d-optimal design*****************
out=0;            %preset 'out' to zero
count=0;          %preset to zero
for a=1:ni
    dopt{8,1}=a;          %stores a value for use in other functions
    iterd=a;            %for graphs and output.
        if flag2(2)==1
                [pointd,XY4,Z4]=optgraph(dopt{5,1}, dopt{5,1}, step1,modeld{2,dopt{9,1}(a)}(a,:),
                   {modeld{:,dopt{9,1}(a)}},{dopt{:,1}},modeld{17,dopt{9,1}(a)}(a), 3);
                [dout{a},dopt{16,1}(a,:),dopt{10,1}(a)]=box1({XY4{:}},Z4,{dopt{:,1}},{modeld{:,
                   dopt{9,1}(a)}},1);
                [LB1,UB1]=gridint(dopt{16,1}(a,:),optrange,step1);
                [dopt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modeld{2,dopt{9,1}(a)}(a,:),{model
                   {:,dopt{9,1}(a)}},{dopt{:,1}},modeld{17,dopt{9,1}(a)}(a),1),dopt{16,1}(a,:)
                   ,[],[],[],[],LB1,UB1);
        else
    %first, find best starting point for optimization using optgraph
                [point(a,:),XY5, Z5(:,:,a)]=optgraph(dopt{11,1}, dopt{12,1}, step1, modeld{2,dopt
                   {9,1}(a)}(a,:),{modeld{:,dopt{9,1}(a)}}, {dopt{:,1}},modeld{17,dopt{9,1}(a)}(a)
                   ,1);
                [dopt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modeld{2,dopt{9,1}(a)}(a,:),{modeld
                   {:,dopt{9,1}(a)}},{dopt{:,1}},modeld{17,dopt{9,1}(a)}(a),1),point(a,:)
                   ,[],[],[],[],[dopt{11,1}(1) dopt{12,1}(1)],[dopt{11,1}(2) dopt{12,1}(2)]);
        end
    %generate new data point
    newdat=MHF(dopt{6,1}(a,:),rep,dopt{3,1});
    %add to existing dataset
    dopt{2,1}=[dopt{2,1};newdat];
    for i=1:mM
        %calculate metrics for each model
      [modeld{2,i}(a+1,:),modeld{3,i}(a+1),modeld{5,i}(a+1,:),modeld{6,i},modeld{7,i}(a+1),modeld
         {8,i}(a+1),modeld{9,i}(a+1),modeld{10,i}(a+1),modeld{11,i}(a+1),modeld{12,i}(a+1),modeld
         {15,i}(a+1),modeld{13,i}(a+1),modeld{16,i}(a+1,:),modeld{17,i}(a+1),modeld{18,i}(a+1),
         residD{i,a},dopt{1,1}(a+1)]=metrix3({modeld{:,i}},{dopt{:,1}},step2);
      allP(i)=mprob(iprob,{modeld{:,i}}, {dopt{:,1}});      %calculate probabilities for each model
    end
    %calculate normalized probabilities for each model and find most
    %probable model
    [P,dopt{9,1}(a+1)]=maxP(allP);
    for i=1:mM
```

```
        %assign probabilities to the respective model arrays
        modeld{4,i}(a+1)=P(i);
    end
    exper=size(dopt{2,1});      %calculates how many experiments have been run
    [out,count,dopt{14,1}(a,:)]=SC2({modeld{:,dopt{9,1}(a+1)}},dopt{8},goal,count,exper(1),rep,
      dopt{17});
        if out==2
                if flag2(4)==1
                        break
                else
                        out=0;          %if SC is not being run, change out to zero so box
                          algorithm can be run
                end
        end
end
clear point
%% For loop for G-optimal design
out=0;          %preset 'out' to zero
count=0;        %preset to zero
for a=1:ni
     gopt{8,1}=a;         %stores a value for use in other functions
     iterg=a;           %for graphs and output.
    if flag2(3)==1
        [pointg,XY4,Z4]=optgraph(gopt{5,1}, gopt{5,1}, step1,modelg{2,gopt{9,1}(a)}(a,:), {modelg
          {:,gopt{9,1}(a)}},{gopt{:,1}},modelg{17,gopt{9,1}(a)}(a), 3);
        [gout{a},gopt{16,1}(a,:),gopt{10,1}(a)]=box1({XY4{:}},Z4,{gopt{:,1}},{modelg{:,gopt{9,1}(a
          )}},0);
        [LB1,UB1]=gridint(gopt{16,1}(a,:),optrange,step1);
        [gopt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modelg{2,gopt{9,1}(a)}(a,:),{model{:,gopt
          {9,1}(a)}},{gopt{:,1}},modelg{17,gopt{9,1}(a)}(a),0),gopt{16,1}(a,:) ,[],[],[],[],LB1,UB1
          );
         else
                %G-optimal designs the next experiment at the point of maximum
                %prediction variance
                %first, find best starting point for optimization using optgraph
                [point(a,:),XY, Z(:,:,a)]=optgraph(gopt{11,1}, gopt{12,1}, step1, modelg{2,gopt
                  {9,1}(a)}(a,:),{modelg{:,gopt{9,1}(a)}}, {gopt{:,1}},modelg{17,gopt{9,1}(a)}(a)
                  ,0);
                 %finish optimization using fmincon
                [gopt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modelg{2,gopt{9,1}(a)}(a,:),{model
                  {:,gopt{9,1}(a)}},{gopt{:,1}},modelg{17,gopt{9,1}(a)}(a),0),point(a,:)
                  ,[],[],[],[],[gopt{11,1}(1) gopt{12,1}(1)],[gopt{11,1}(2) gopt{12,1}(2)]);
        end
    %generate new data point
    newdat=MHF(gopt{6,1}(a,:),rep,gopt{3,1});
    %add to existing dataset
    gopt{2,1}=[gopt{2,1};newdat];
    for i=1:mM
        %calculate metrics for each model
      [modelg{2,i}(a+1,:),modelg{3,i}(a+1),modelg{5,i}(a+1,:),modelg{6,i},modelg{7,i}(a+1),modelg
        {8,i}(a+1),modelg{9,i}(a+1),modelg{10,i}(a+1),modelg{11,i}(a+1),modelg{12,i}(a+1),modelg
        {15,i}(a+1),modelg{13,i}(a+1),modelg{16,i}(a+1,:),modelg{17,i}(a+1),modelg{18,i}(a+1),
        residG{i,a},gopt{1,1}(a+1)]=metrix3({modelg{:,i}},{gopt{:,1}},step2);
      allP(i)=mprob(iprob,{modelg{:,i}}, {gopt{:,1}});     %calculate probabilities for each model
    end
    %calculate normalized probabilities for each model and find most
    %probable model
    [P,gopt{9,1}(a+1)]=maxP(allP);
    for i=1:mM
        %assign probabilities to the respective model arrays
        modelg{4,i}(a+1)=P(i);
    end
    exper=size(gopt{2,1});       %calculates how many experiments have been run
    [out,count,gopt{14,1}(a,:)]=SC2({modelg{:,gopt{9,1}(a+1)}},gopt{8},goal,count,exper(1),rep,
      gopt{17});
    if out==2
            if flag2(4)==1
        break
else
        out=0;            %if SC is not being run, change out to zero so box algorithm can be run
end
    end
 end
clear point
%% For loop for p-optimal design
     out=0;            %preset 'out' to zero
count=0;          %preset to zero
for a=1:ni
        popt{8,1}=a;          %stores a value for use in other functions
        iterp=a;            %for graphs and output.
        %P-optimal
        if flag2(1)==1
                [pointp,XY4,Z4]=optgraph(popt{5,1}, popt{5,1}, step1,modelp{2,popt{9,1}(a)}(a,:),
                  {modelp{:,popt{9,1}(a)}},{popt{:,1}},modelp{17,popt{9,1}(a)}(a), 3);
                [pout{a},popt{16,1}(a,:),popt{10,1}(a)]=box1({XY4{:}},Z4,{popt{:,1}},{modelp{:,
                  popt{9,1}(a)}},2);
                [LB1,UB1]=gridint(popt{16,1}(a,:),optrange,step1);
                [popt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modelp{2,popt{9,1}(a)}(a,:),{modelp
                  {:,popt{9,1}(a)}},{popt{:,1}},modelp{17,popt{9,1}(a)}(a),2),popt{16,1}(a,:)
                  ,[],[],[],[],LB1,UB1);
        else
        %first, find best starting point for optimization using optgraph
        [point(a,:),XY3,Z3(:,:,a)]=optgraph(popt{11,1}, popt{12,1}, step1, modelp{2,popt{9,1}(a)}(
          a,:),{modelp{:,popt{9,1}(a)}}, {popt{:,1}},modelp{17,popt{9,1}(a)}(a),2);
        %finish optimization using fmincon
        [popt{6,1}(a,:),fnew] = fmincon(@(xd)optfun(xd,modelp{2,popt{9,1}(a)}(a,:),{modelp{:,popt
          {9,1}(a)}},{popt{:,1}},modelp{17,popt{9,1}(a)}(a),2),point(a,:) ,[],[],[],[],[popt
          {11,1}(1) popt{12,1}(1)],[popt{11,1}(2) popt{12,1}(2)]);
    end
        %generate new data point
    newdat=MHF(popt{6,1}(a,:),rep,popt{3,1});
    %add to existing dataset
    popt{2,1}=[popt{2,1};newdat];
    for i=1:mM
        %calculate metrics for each model
      [modelp{2,i}(a+1,:),modelp{3,i}(a+1),modelp{5,i}(a+1,:),modelp{6,i},modelp{7,i}(a+1),modelp
        {8,i}(a+1),modelp{9,i}(a+1),modelp{10,i}(a+1),modelp{11,i}(a+1),modelp{12,i}(a+1),modelp
        {15,i}(a+1),modelp{13,i}(a+1),modelp{16,i}(a+1,:),modelp{17,i}(a+1),modelp{18,i}(a+1),
        residP{i,a},popt{1,1}(a+1)]=metrix3({modelp{:,i}},{popt{:,1}},step2);
      allP(i)=mprob(iprob,{modelp{:,i}}, {popt{:,1}});      %calculate probabilities for each model
    end
    %calculate normalized probabilities for each model and find most
    %probable model
    [P,popt{9,1}(a+1)]=maxP(allP);
    for i=1:mM
        %assign probabilities to the respective model arrays
        modelp{4,i}(a+1)=P(i);
    end
    exper=size(popt{2,1});      %calculates how many experiments have been run
    [out,count,popt{14,1}(a,:)]=SC2({modelp{:,popt{9,1}(a+1)}},popt{8},goal,count,exper(1),rep,
      popt{17});
    if out==2
           if flag2(4)==1
        break
else
        out=0;          %if SC is not being run, change out to zero so box algorithm can be run
end
    end
end
clear point
%save data to file
    p=datestr(now,'mm_dd_yy');
%    nwkdir=[wkdir,p,'\',p];       %updates directory with today's date
    ch7=[p,'rep', num2str(num),'.txt'];
    ch5=[p,'_',num2str(num)];
    save(ch5)
    %------------------------------------
%% Save all figures
SaveAllFigures(ch5,1,num);
close all       %this command closes all graphs
end

%% extra functions needed
%%%%%%%%%%%%%%%%%%%%%%%%%%box1%%%%%%%%%%%%%%%%
function [out,point,boxmax]=box1(mat,Z,opt,model8,flag)
%this function takes as input the matrix of x and y values 'X', and the
%matrix of some functions evaluations in the (x,y) space, Z.  'boxmax' is the
%maximum allowed value calculated from avareage output + CI
counter=0;      %need something to track how many points are below boxmax
a=opt{8};       %iteration
theta=model8{2}(a,:);   %parameters
sigsq=model8{17}(a);    %exp error
hand=model8{1};         %handel for model function
d=opt{1}(a);            %max value of data for Jac.m
J=model8{6};            %X matrix (derivative of model wrt parameters)
o=length(opt{2});       %calculates how many experiments have been performed
p=model8{14};           %# parameters in model
CI=model8{18}(a);       %this is the confidence interval
value=model8{12}(a);    %value is the value at the est. opt point of the most probable model
X=mat{1};       Y=mat{2};
boxmax=value+CI;                %this assumes that one is trying to minimize, meaning one should
   try to find a value BELOW boxmax
for i=1:length(X)
    for j=1:length(X)
        point=[X(i,j) Y(i,j)];
        %calculate prediction variance at point
        Jopt=Jac(theta,hand,point,d);
        PVopt1=PV(J,Jopt,sigsq);
        %calculate confidence interval at estimated optimal point.  Equation taken
        %from pg 395 Montgomery.  **Jopt should be a 3x1 matrix, so the transpose is
        %reversed**
        CI2=tinv(0.975,(o(1)-p))*sqrt(sigsq*(Jopt*inv(J'*J)*Jopt'));
        if Z(i,j) < boxmax || (Z(i,j)-CI2) < boxmax      %want value to be less than boxmax since
          the point is to minimize output
                counter=counter+1;
                out(counter,:)=[Z(i,j) CI2 X(i,j) Y(i,j)];       %the column indice of Z is the x
                  indice, row is for y indice
        end
    end
end
if counter==0           %if nothing was found that met contstraints, end function
    out=1;              %out returns a dummy variable so no error
    return
end
%out is of the form [output CI x y] for as many rows as values below 'boxmax'
% part 2 of function-picking best value for optimality
[n,m]=size(out);
if flag==3
        point=[out(1,3) out(1,4)];      %if running random, don't calculate optimal value
else
        for k=1:counter        %calc value of optimality (depends on flag) at the points below
          Pvalue
                out(k,5)=optfun([out(k,3) out(k,4)],model8{2}(opt{8},:),model8,opt,model8{17}(opt
                  {8}),flag);
        end
        %-----------FINDING MINIMUM--------------------
        [val,I]=min(out(:,5));          %finds minimum of each column in 'b' and its indice
        xfin=out(I,3);
        yfin=out(I,4);
        point=[xfin yfin];        %index of minimum point
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Euclid%%%%%%%%%%%%%%%
function dist=Euclid(X,Y)
%This is a function that finds the euclidian distance between two points
%X=[x1 y1]  is first point
%Y=[x2 y2]  is second point
first=(X(1)-Y(1))^2;
second=(X(2)-Y(2))^2;
dist=sqrt(first+second);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Gridint%%%%%%%%%%%%%%
function [LB1, UB1]=gridint(point,range,step1)
%function calculates the ranges to use to optimize around the point from grid algorithm
perc=0.80;                                          %represents the percentage of interval one wants
   to use for bounds
interval=(range(2)-range(1))/step1;        %calculates step between grid points
point1=point(1);          point2=point(2);         %stores components
LB1(1)=point1-perc*interval;
```

```
if LB1(1)<range(1), LB1(1)=range(1); end          %if LB is outside of experimental bound, than make
    lower bound the experimental boundary
UB1(1)=point1+perc*interval;
if UB1(1)>range(2), UB1(1)=range(2); end          %if UB is outside of experimental bound, than make
    upper bound the experimental boundary
LB1(2)=point2-perc*interval;
if LB1(2)<range(1), LB1(2)=range(1); end          %if LB is outside of experimental bound, than make
    lower bound the experimental boundary
UB1(2)=point2+perc*interval;
if UB1(2)>range(2), UB1(2)=range(2); end          %if UB is outside of experimental bound, than make
    upper bound the experimental boundary
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Jac%%%%%%%%%%%%
function z=Jac(theta,model1,data,d)
%This function calculates the jacobian for each model w.r.t. the parameter
%and given the input data.  It calculates the derivative by changing one of
%the parameters by a small amount, and seeing how that changes the value of
%the output.  (i.e. Finite Difference Method)
%-----------------------------------------------------
%list of variables
%theta= best fit parameters
%cell=cell array with model data
%data=experimental data matrix
%d=maximum value of y(exp) (in nucleation study, this is data(:,3) from run_simxx for calculation
   of an appropriate dsig
%n, m, p,r,s, S,t, thetahat,x,z
%-------------------------------------------------
dsig = 0.001*d;   %deltatheta (for derivative calculation)
[n,nC]=size(data);            % n is the number of rows contained in data
 for s=1:length(theta)
     dtheta=theta;      %storing theta values to be changed
     dtheta(s)=dtheta(s)+dsig;   %adds deltatheta to appropriate parameter #
    for p=1:n
         x=[data(p,1) data(p,2)];          %saves data in vector 'x'
         r=feval(model1,x,theta);          %calculating function value without dsig
         t=feval(model1,x,dtheta);   %calculates function with dsig
        z(p,s) = (t-r)/dsig;                 %calculates derivative (change in function value)/(
          change in parameter)
     end
 end
end
%%%%%%%%%%%%%%%%%%%%%%%%maxP%%%%%%%%%%%%%%%%%%%
function [Pnorm,I]=maxP(P)
m=size(P);
Pnorm=zeros(m,1);
psum=0;
%normalize probabilities
for j=1:m
    psum=psum+P(j);
end
for k=1:m
    Pnorm(k)=P(k)/psum;      %normalizes and outputs probability to screen
end
%find model with largest probability
[Pmax,I]=max(Pnorm);
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%metrix3%%%%%%%%%%
function [theta,Sr,point,J,Dopt,PVx1,PVopt1,PVopt2,err1,pred,dist,pred2,F,sigmasq,CI,r,d]=metrix3(
   model2,opt,step1)
%This function will reduce the clutter in the main function.  This function
%will calculate the D-optimal value
%%%%INPUT Definitions
%model2=array passed with needed information for optimality
%err=experimental error
%range=lower and upper bounds for calculating new thetas
%optrange=range for finding optimum point
%step1=how many steps one wants optgraph to perform
%constants for MHF
%%%%OUTPUT Definitions
%theta=predicted thetahat for this iteration
%Sr=predicted Srhat for this iteration
%d=predicted max value of y from data
%J=predicted jacobian for this iteration
%data=data matrix plus new experiment from this iteration
%Dopt=D-optimal value for this iteration
%PVx1=prediction variance for experimental point
%pred=predicted value at estimated optimal point
%err1=error of predicted value to actual value at real optimal point
%PVopt1=PV at initial estimated optimal point
%PVopt2=PV at real optimal point
%pred2=output from system
%F= F statistical values for LOF (1) is for model (2) is F for comparison
     %to (1)
%sigmasq=experimental variance calculated from experimental data
%optpoint=estimated optimal point for this iteration (after new experiment
%is generated (used in calculations for next round
pointopt=[-3.8 -3.32];  %coordinates of optimal point in MHF
optpt=43.3;                %value at optimal point
%calculate new thetahat and Srhat
if model2{14}<3          %this was added for any model that has less than 3 parameters
          LBguess=opt{4}(1,1:2);   UBguess=opt{4}(2,1:2);
else      LBguess=opt{4}(1,:);     UBguess=opt{4}(2,:);
end
[theta, Sr, G] = newfit(opt{2}, model2{1},model2{2}(opt{8},:),[LBguess; UBguess]);
n=size(opt{2});
sigmasq=Sr/(n(1)-model2{14});                %calculating experimental variance
d=max(round(opt{2}(:,3)));                   %calculates maximum of output for jacobian calculation
J=Jac(theta,model2{1},opt{2},d);    %calculates Jacobian of ALL the data to use in D-optimal.
%find optimal point predicted by the optimality
%first, use optgraph to start optimization
[pointp,XY4,Z4]=optgraph(opt{5}, opt{5}, step1, theta, model2,opt,sigmasq, 3);
[point]=fmincon(@(point)feval(model2{1},point,theta),pointp,[],[],[],[],[opt{5}(1) opt{5}(1)],[opt
   {5}(2) opt{5}(2)]);
%Prediction Variance at new point
Jnew=Jac(theta,model2{1},opt{6}(opt{8},:),d);
PVx1=PV(J,Jnew,sigmasq);
%Calculate prediction at estimated optimal point
pred=feval(model2{1},point,theta);      %from model
sysout=feval(@MHF,point,1);          %from system
pred2=sysout(3);
```

```
err1=(pred-pred2)^2;                    %error calculation at estimated optimal
                                         %point
%calculate prediction variance at estimated optimal point
Jopt=Jac(theta,model2{1},point,d);
PVopt1=PV(J,Jopt,sigmasq);
%calculate confidence interval at estimated optimal point.  Equation taken
%from pg 395 Montgomery.  **Jopt should be a 3x1 matrix, so the transpose is
%reversed**
CI=tinv(0.975,(n(1)-model2{14}))*sqrt(sigmasq*(Jopt*inv(J'*J)*Jopt'));
%calculate prediction variance at real optimal point
Jopt=Jac(theta,model2{1},pointopt,d);
PVopt2=PV(J,Jopt,sigmasq);
%calculate final D-optimal value using function 'dfun'
Dopt=optfun(opt{6}(opt{8},:),theta, model2,opt,sigmasq,1);
%calculate distance between real optimum point and estimated optimum point
dist=Euclid(pointopt,point);
%calculate the lack of fit F statistics
F=50;%
%calculate studentized residuals
[Sr2,resid]=newerr(theta,opt{2},model2{1});  %finds residuals from newerr
Hat=J*(J'*J)^(-1)*J';    %calculates hat matrix
for i=1:n(1)
    r(i)=resid(i)/sqrt(Sr*(1-Hat(i,i)));
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%MHF%%%%%%%%%%%
function data=MHF(X,rep2,err)
data=zeros(rep2,3);
%This is the Modified Himmelblau function as a test surface
X1=X(1);
X2=X(2);
p=[-11 -7 1 3 57];        %these are the parameters to be changed
if nargin==2
     err=0;
end
for kj=1:rep2
y=(X1.^2+X2+p(1)).^2+(X1+X2.^2+p(2)).^2+p(3).*X1+p(4).*X2+p(5)+normrnd(0,err);
data(kj,:)=[X1 X2 y];  %Stores the data and experimental settings together
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%mprob%%%%%%%%%%
function [P]=mprob(iprob,model3, opt)
%This function calculates the Bayesian probability for a particular model
%given the initial probability, # parameters, Srhat, and # repetitions
     P=iprob*2^(-model3{14}/2)*model3{3}(opt{8})^(-opt{7}/2);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%newerr%%%%%%%%%
function [Sr,Perr] = newerr(theta,data,fmodel)
%--------------------------------------------
%list of variables
%theta is model parameters
% data in [x y] form
%fmodel is handle to model function
%n,nc,Perror,yexp, y, Sr,i
%Perr=residuals for each datapoint
```

```
%-----------------------------------------------
[n,nc] = size(data);          %gets size of data matrix
Perror = zeros(n,1);          %initializes Perror to zero
for i = 1:n
    yexp=data(i,3);           %gets experimental result
    y=feval(fmodel,[data(i,1) data(i,2)],theta);     %calculates result from model
    Perr(i)=yexp-y;           %this is the residual
    Perror(i) = (Perr(i))^2;  %calculates error of model prediction
end
Sr = sum(Perror); %sums up Perror to get mean square error (Sr)
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%newfit%%%%%%%%%
function [theta,Sr,G] = newfit(data,fmodel,po, range)
%function calculates the best fit parameters to the data , theta, as well as the
%model error, Sr
% data in [y x] form
%model is the function handle
%po is the parameter guess
%range contains the bounds for the optimization
%Sr0 is initial error
%theta, Sr, G, output are outputs of fmincon
%-----------------------------------------------------
Sr=0; Sr0=0;
Sr0 = newerr(po,data,fmodel); % Initial error in prediction
% Minimize over the fit
%setting tolerance for fmincon
OPTIONS = optimset('DiffMinChange',1e-6,'Display', 'iter','Diagnostics', 'on', 'LargeScale', 'off'
    );%'TolX',sett(1,1),'TolFun',sett(1,1),'TolCon',sett(1,1),  'MaxFunEvals', sett(2,1),'
    Diagnostics', 'on', 'LargeScale', 'off');
%'Display', 'iter',              --this could go back in later
[theta,Sr,G,output] = fmincon(@(theta)newerr(theta,data,fmodel),po,...
                          [],[],[],[],range(1,:),range(2,:));
Sr;
%G is exitflag of fmincon
%output gives information on optimization
%G, output used for debugging
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%optfun%%%%%%%%%%%%%%%
function z=optfun(xd,theta, model4,opt,sigmasq2,flag)
%%%This is a function for optimal design.  Depending on the number of
%inputs, the function will do either d-, g-, or p- optimal design.
%------------------List of variables-------------
%model4=cell array for model being investigated
%opt=cell array for optimality being run
%flag=tells function to run either
    %0 = gfun
    %1=dfun
    %2=pfun
%first, need to find jacobian for new experimental point
J1=Jac(theta,model4{1},xd,opt{1}(1));
X=[model4{6}; J1];
if flag ==0
    %do gfun-tries to reduce PV over entire experimental range by picking
    %point with maximum PV and running next experiment at that point
    z = -J1*inv(model4{6}'*model4{6})*J1'*sigmasq2;
```

```
elseif flag ==1
    %calculates D-optimality (maximizes det(X'*X) )
    z = -det(X'*X);
elseif flag==2
   %do pfun -pfun uses optimal point and experimental point to find the
   %prediction variance at the optimal point.
   %will find the experimental point with the largest prediction variance
   %and run more experiments at that point when run with fmincon
    %compute jacobian for optimal point 'xp'
    Jnew=Jac(theta,model4{1},model4{5}(opt{8},:),opt{1}(1));
    %compute prediction variance
    z = Jnew*inv(X'*X)*Jnew'*sigmasq2;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%optgraph%%%%%%%%%
function [point,coordin,Z]=optgraph(xrange, yrange, num, theta, model5, opt,sigmasq3,flag)
%the purpose of function is to quickly graph prediction variance to
%find starting point for fmincon optimization
%--------------------------LIST OF VARIABLES---------
%xrange=range of variable x [min max]
%yrange=range of variable y  [min max]
%num=number of points desired in table
%data=raw data
%point is output that contains indices of minimum point [x indice, y indice]
%d, I, m1,m2, score, sig, val, vari, xd,X, xfin, yfin,yd,Y,z,Z
%------------------ SETTING UP GRID---------------
xd=(xrange(2)-xrange(1))/num;        %calculates step between x-values
yd=(yrange(2)-yrange(1))/num;        %calculates step between y-values
x=xrange(1):xd:xrange(2);
y=yrange(1):yd:yrange(2);
[X,Y]=meshgrid(x,y);                %makes grid for calculation (used for graphing)
% Calculating values
Z=zeros(length(X),length(Y));
 for i=1:length(X)
    for j=1:length(Y)
        if flag==3
            %this is the functional form to run the model to find the
            %optimal point. Note when this is used, 'fun1' is just a
            %placeholder and not used in 'optgraph.m'
            Z(i,j)=feval(model5{1},[X(i,j) Y(i,j)],theta);
        else
            Z(i,j)=optfun([X(i,j) Y(i,j)],theta,{model5{:}},{opt{:}},sigmasq3,flag);
        end
    end
 end
coordin={X,Y};
        %-----------FINDING MINIMUM----------------
[val,I]=min(Z);           %finds minimum of each column in 'b' and its indice
[score, I2]=min(val);    %finds minimum of vector val and its indice
xfin=x(I2);
yfin=y(I(I2));
point=[xfin yfin];         %index of minimum point
%%%--------------------------------------------------
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PV%%%%%%%%%%%%
```

```
function z=PV(J,J1,err)
%This function calculates the prediction variance given a Jacobian, 'J',
%the experimental point variance, 'J1', and the experimental error, 'err'
z=J1*inv(J'*J)*J1'*err;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%SaveAllFigures%%%%%%
function SaveAllFigures(dir,z,num,filetype)
%if one doesn't specify filetype, autoset to fig
if nargin < 4,  filetype = 'fig'; end
%getting info for each figure-taken from internet
ChildList = sort(get(0,'Children'));
for cnum = 1:length(ChildList)
if strncmp(get(ChildList(cnum),'Type'),'figure',6)
saveas(ChildList(cnum), [dir,'_', num2str(z),'_',num2str(num), '_', num2str(ChildList(cnum)), '.'
  filetype]);
end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%SC2%%%%%%%%%%%%%
function [out,count,flag3]=SC2(model9,opt,goal,count,N,rep,tol)
%This is a function for the stopping criterion for simulations.  The values
%for 'model9' should be coming from MOST PROBABLE model
%'goal' is the desired confidence interval on prediction
%'N' is # experiments
%ni  is # repititions
%tol=tolerance for change in parameters
flag=1;          %flag for thetas should be initially set to 1
flag1=1;         %flag for srhat vs tol(2)
flag2=1;         %flag for CI vs goal
out=0;           %initialize 'out'  0== continue experiments
%------------assigning values from input arrays----
mark=opt+1;     %opt{8} is the number of repititions, need to add one
                     %since original experiments not counted in this #
thetanew=model9{2}(mark,:);      %stores values of new thetas
thetaold=model9{2}(mark-1,:);    %stores old thetas
Srnew=model9{3}(mark)/N;             %new Srhat (normalized by # experiments)
Srold=model9{3}(mark-1)/(N-rep);             %old Srhat (normalized by # experiments)
noise=model9{17}(mark);                   %estimate of experimental error (noise)
CI=model9{18}(mark);                %conf. intervals for model
n=length(thetanew);                 %find # parameters
compa=zeros(n,1);               %preallocating memory for 'compa'
%-------------------------theta comparison---------
for i=1:n
        %first, calculate each comparison for thetas
        compa(i)=abs((thetaold(i)-thetanew(i))/thetaold(i));
        if compa(i) > tol
                flag=0;           %if parameters are changing, set flag =0
                        %if at least one parameter is changing, experiments
                        %should continue.  If none are changing, then next
                        %step and flag=1
        end
end
%-------------------------Srhat comparison------------
compa1=abs((Srold-Srnew)/Srold);
if compa1 > tol, flag1=0;       %if Srhat is changing, continue experiments (i.e. flag=0)
```

```
end
%-----------confidence interval comparison--------------
if CI > goal, flag2=0;             %if CI has not reached goal, keep doing experiments;
end
%-------------------------end comparison-------------
%stopping criterion are 'thetas AND Srhat not changing' OR 'CI < noise
%level' OR 'CI < desired CI'
if flag1==1 || flag2==1, out=1; count=count+1; else count=0;          %if continuing experiments, '
   count' should equal 0
end
flag3=[flag flag1 flag2];         %flag3 tells user result of each comparison
if count==2           %if count=2, then optimal point was just run twice in a row.  If it's not
   changing much
                      %no need to continue experiments
     out=2;           %out=2 means to stop experiments
end
end
%% model 2%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function y=mod2(X,p)
%This is the first model, all correct
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);

y=p(1)*X1^4+X2^4-21*X1^2 +2*X1^2*X2+p(2)*X1*X2^2-13*X2^2-13*X1-19*X2+p(3);
end
%% model 6
function y=mod6(X,p)
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)

X1=X(1);
X2=X(2);
                    %X1^2 term incorrect
y=X1^4+X2^4-10*X1^2 +2*X1^2*X2+p(1)*X1*X2^2-13*X2^2-13*X1+p(2)*X2+p(3);
end

%% model 7
function y=mod7(X,p)
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);
                    %x1^2 term incorrect                %x2^2 term incorrect
y=X1^4+X2^4-10*X1^2 +2*X1^2*X2+p(1)*X1*X2^2-1*X2^2-13*X1+p(2)*X2+p(3);
end

%% model 8
function y=mod8(X,p)
%X is the experimental setting
```

```
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);
                                                                            %X2 term
                                                                            %missing
y=X1^4+X2^4-21*X1^2 +2*X1^2*X2+p(1)*X1*X2^2-13*X2^2-13*X1+p(2);
end

%% model 9
function y=mod9(X,p)
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);
                                                                     %X1 term missing
y=X1^4+X2^4-21*X1^2 +2*X1^2*X2+p(1)*X1*X2^2-13*X2^2+p(2)*X2+p(3);
end
%% model 10
function y=mod10(X,p)
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);
                                                          %X1 and X2 terms missing
y=p(1)*X1^4+X2^4-21*X1^2 +2*X1^2*X2+p(2)*X1*X2^2-13*X2^2+p(3);
end
%% model 11
function y=mod11(X,p)
%X is the experimental setting
%p is a vector of the model parameters (currently 3 parameters)


X1=X(1);
X2=X(2);
                    %X1, X2, X1^2, and X2^2 terms missing
y=p(1)*X1^4+X2^4+2*X1^2*X2+p(2)*X1*X2^2+p(3);
end
```

Listing D.1: trial23.m in Matlab

D.2 Matlab code for film growth simulation study in Matlab

This is the file used for the film growth simulation study.

```
function ni=Dsim29(nameout,err,flag2,rep,sctol,num)
%5/8/2008      keeps changes to Dsim27, but mod_mech and mod_mech2 are modified to make them
    better, to make models more competitive
%This function is corrected to include CI in box calculation
```

```
%This is a function to run the simulation code, and takes the names of the
%output file and number of repititions.  This is the first attempt to use the
%optimalities with the nucleation models.  Also incorporates F(T,C)
%*******List of inputs and outputs************
%nameout= name of output file for reports
%err= desired level of noise to be added to system
%flag2= vector of length 4 for each optimality, tells whether or not to run box algorithm [P D G
   Run] 'Run' is a flag to say whether to run D and G or not, 1=yes
%rep= # repititions to perform for each simulated experiment
%num= # to help differentiate between different simulations
%Srnorm= value of normal Sr (kind of a dummy output)
%List of variables******************************
%a, avg, b,cells,  ch2, ch4, col1, col2, count,d, data, deriv, discr, emp, err,
%err1, exp, F, flag, flag2,FVAL, G, guess,h,i, I, I2, iter, ik,j,k, K, kk,L, Low1, m, mC, mech,
   mod, model,
%n, namein, nameout, nC, ni, num, Options, other,output, p, pM, pMcond, psum, P,
%q, Q, r, R, raw, rep, s,sc, simdata, Srhat, SrLS, SrLSnorm
%scal, scal2, set, sett, step, step2,thetahat, T,tol, u, Up1, v, ve, x, XO, Xset, Xsetnew, z
%**********************************************
%Check Validity of inputs
if ischar(nameout)==0
    error('nameout needs to be a character string, use single quotes')
end
%---------------------USER INPUTS-------------
source='Dsim29';          %Keeping track of what generator file generated what data
goal=5;            %defines desired magnitude of confidence interval (1 second)
flag=0;                 %flag==1 enables graphs at end
ni=16;              %max # of iterations
Low1=[873; 0.3];  %lower bounds for experimental settings
Low2=scale(Low1,0);     %scales lower bounds
Up1=[1073; 1.5];  %upper bounds for experimental settings
Up2=scale(Up1,0);       %scales upper bounds
step1=15;                      %number of steps to take when making grid of design space for 1st
   part of discrim function
step2=20;                    %finding optimal point, needed more spots on grid
cells=[19 17];       %defines size of cell arrays for opt and model array
XO=[971 0.5]; %XO=initial experimental settings for fmincon discrim func;
Xset(1,:)=scale(XO,0);  %scales XO (set of experimental settings, [T C]
%err=1e-4;               %variance of gaussian noise added to data during simulation
tol=1e-18;          %Tolerance for fmincon
tol2=1e-10;          %Tolerance for TolX
iterate=100;          %iterations for fmincon
sett=[tol tol2;iterate 0];        %puts tolerances and scaling in a vector for fmincon
sc=1e6;            %scaling factor for error calculation
if sc==0           %scal cannot be =0, or error calc won't work
    error('scal must be > 0');
end
sc2=1%e17;       %this is a scaling factor for fmincon search on optfun
Pvalue=zeros(ni,1);
rand('state',sum(100*clock));           %need to initialize seed for random number generator
repeat=1;               %determines how many times to do simulations
%-----------------------Make model arrays---------
%this makes the model arrays with information about each model
%models are organized by column
% model(1,:)=name of model
% model(2,:)=%# of parameters in the model
% model(3,:)=model{3,1}=[.7 .1 .7];            %initial guess for emp model parameters
        %initial guess for model parameters
% model(4,:)=Lower bound for model parameters
% model(5,:)=Upper bound for model parameters
% model{6,:}=Flag (==0, no derivative  ==1 derivative in model{8,:}
% model{7,:}=Flag for Compfit (==0, normal   ==1, for mod_avg)
                % available data points

%************************% Empirical model*********
model{1,1}=@mod_emp;                %name of model to be loaded
model{3,1}=[.7 .1 .3];              %initial guess for emp model parameters
model{2,1}=length(model{3,1});                     %# of parameters in the model
model{4,1}=-300*ones(1,3);          %Lower bound for emp model
model{5,1}=300*ones(1,3);            %Upper bound for emp model
model{6,1}=0;
model{7,1}=0;
%%************************%Mechanistic model*******
model{1,2}=@mod_mech;               %name of model to be loaded
model{3,2}=[0];                 %initial guess for model parameters
model{2,2}=length(model{3,2});                     %# of parameters in the model
model{4,2}=[-30];               %Lower bound for model
model{5,2}=[20];            %Upper bound for model
model{6,2}=0;
model{7,2}=0;
%%************************Average model************
 model{1,4}=@mod_avg;               %name of model to be loaded
 model{3,4}=1;                  %initial guess for model parameters
 model{2,4}=length(model{3,4});       %# of parameters in the model
 model{4,4}=-50;                %Lower bound for model
 model{5,4}=50;             %Upper bound for model
 model{6,4}=1;
 model{7,4}=1;
%************************SECOND Mechanistic model***
model{1,3}=@mod_mech2;              %name of model to be loaded
model{3,3}=[50 50];                 %initial guess for model parameters
model{2,3}=length(model{3,3});      %# of parameters in the model
model{4,3}=[-20 -20];               %Lower bound for model
model{5,3}=[200 100];           %Upper bound for model
model{6,3}=0;
model{7,3}=0;
[m,mC]=size(model);          % mC is the number of models contained in models.mat
%----------------------make opt arrays----------
%opt cell arrays store info as follows:
%{1}=d                       max value of experimental data
%{2}=data                    stores the simulated data
%{3}=err                     stores error
%{4}= optrange               range for experimental points (upper and lower
        %bounds)
%{5}=experimental point      next experimental point to be run
%{6}= # repetitions to be run
%{7}= element to pass value of 'a' to other functions
%{8} stores number of correct model for each iteration
%{9} stores max value acceptable for box calc
%{10}=system output for stopping criterion
```

```
%{11}=verdict                    tells user why simulation was stopped
                              %1=theta and Srhat not changing
                              %2=CI < sqrt(noise)
                              %3=CI < CI (desired)
%{12}=counter                   counter for stopping criterion, see SC for all
                                %stopping criterion questions
%{13}=sett                    %this has the settings for fmincon
%{14}=scal                    %this carries the scalings to be used for newfit.m and newerr.m
%{15}= stores the points found using box1 and values of function at those points [x1 x2]
%{16}= tolerance for SC (may not be used in growth time simulations..)
%{17}= experimental points descaled
init=feval(@cell,cells(2),1);      %extra cell array for inital calcs
%------------------------------ result arrays----------
%result arrays for each model and optimality
%1=thetahat
%2=srhat
%3=normalized probability of model
%4=estimated optimal point
%{5}=J                           Jacobian of data with respect to parameters
%{6}=D                           D-optimal value of model
%{7}=PVxnew                      prediction variance at new experimental point
%{8}=PVxopt                      prediction variance at estimated optimum point
%{9}=model prediction at estimated optimal point
%{10} holds F statistics for LOF when it is calculated
%{11}=experimental error (calculated)
%{12}=confidence interval on predictions
%{13}=empty
%{14}=model prediction error
%{15}= output from system at optimal point
%{16}=objective function output
%{17}=unscaled estimated optimal point
%{18}=n_isl from model
%{19}=n_isl from true system
result=feval(@cell,cells(1),mC);      %extra cell array for inital calcs
%************************************************
%%-------------------------MAIN CODE--------------------
%************************************************
allP=zeros(mC,1);
iprob=1/mC;                     %initial probability for all models so they're equal
 for z=1:repeat;                  %run experiment different times with a different initial parameter
    guess
    err1=err*ones(1,repeat);                 %variance of gaussian noise added to data during
       simulation
%        for h=1:mC
%             model{3,h}=guess{h}(z,:);     % %reassigns initial parameter estimates
%        end
% --------------------------------%GENERATING DATA----------
% %Initial experiments determined using T, C as process variables and setting
% %up a 2^2 factorial experiment
                    %simdata needs 5 inputs  %x1=T [K]  %x2=C [micromol/L]
                    %x3=# for saving files/housekeeping   (used in 'simdata.m')
                    %x4=variance of gaussian noise added to generated data
                    %T and C are normalized, see values below
                    %simdata outputs two things matrix 'data#' and 'param'
                           %data is a 5 column matrix with [x1 x2 n3last tf max_n1]
```

```
                                        %n3last = final N_isl generated from simdata
                                        %tf =final time (generated data always goes to certain level of coverage
                                        %called 'theta'- see notation on simdata.m-not used in rest of
                                                  %simulation
                                        %max_n1= max number of N_1 from simulation.  kept in case we need to
                                                  %study simdata.m more, not used in rest of simulation
                                  %param are the parameters used in simdata to generate the data, at this
                                        %point, it is always [0 0.8 0.2] corresponding to [E_i E_d
                                        %\sigma]
start1=datestr(now)
                        [data1]=Simdata(Low2, rep,err1(z));        % inputs are T=873K, C=0.01
                        [data2]=Simdata([Up2(1); Low2(2)], rep,err1(z));        % inputs are T=1073K, C=0.01
                        [data3]=Simdata([Low2(1);Up2(2)], rep,err1(z));       % inputs are T=873K, C=1.5
                        [data4]=Simdata(Up2, rep,err1(z));     % inputs are T=1073K, C=1.5
                       %saves all generated data into one data matrix
                        data=[data1(:,1:3);data2(:,1:3);data3(:,1:3);data4(:,1:3)];
                       %raw keeps all columns of the original data for future reference
                        raw=[data1; data2;data3;data4];
                        d=max(round(data(:,3)*100))/100;                 %calculates maximum of output for jacobian calculation.  100 is a factor used to make sure 'd' has enough decimal places and is not rounded to zero
            %preset the opt cell arrays to the correct initial values
            init{2}=data; init{1}=d; init{3}=err; init{4}=[Low2 Up2]; init{5}=Xset; init{6}=rep; init{7}=1;  init{11}=zeros(1,3);   init{13}=sett;
            init{14}=sc;     init{16}=sctol;
 %-------------%PARAMETER FIT AND PROBABILITY CALCULATION-----
                  [n,nC]=size(data);           % n is the number of rows contained in data
                  % Compute the least squares fit parameters (theta1hat)and
                              %error for parameters (Sr1hat) for each model
                              %last number in input for compfit cell array for model
                  for j=1:mC
                              [result{1,j}(1,:),result{2,j}(1),result{4,j}(1,:),result{5,j},result{6,j}(1),result{7,j}(1),result{8,j}(1),result{14,j}(1),result{9,j}(1),result{15,j}(1),result{10,j}(1,:),result{11,j}(1),result{12,j}(1),resid{1},init{1}(1),result{16,j}(1),result{18,j}(1),result{19,j}(1),result{20,j}(1),data10{j,1},ndata{j,1},result{21,j}(1,:),result{22,j}(1),result{23,j}(1)]=metrix3({model{:,j}},{init{:}},{result{:,j}},step2);  %metrix3 calculates parameter fits and other measures of quality for each model-see description of 'metrix3.m'
                              result{17,j}(:,1)=scale(result{4,j}(1,:),1);         %descales T and C
                              model{3,j}=result{1,j}(1,:);          %replaces initial guess with new theta for next iteration
                              pMcond(j) = mprob(iprob,model{2,j},init{6},result{2,j});  % Compute the conditional probabilities of the models
                  end
            % normalize probabilities and find most probable
                        [P,init{8}]=maxP(pMcond');      %normalizes and outputs probability to screen
                        for zz=1:mC
                                    result{3,zz}=P(zz);      %stores norm. prob. for each model
                        end
%---------assigning arrays for each optimality-------
popt=init;presult=result;        dopt=init;gopt=init;dresult=result; gresult=result;praw=raw;draw=raw;graw=raw;%assign cell arrays to correct optimality for storage%%
%%       loop for pfun********************************
```

```
fprintf('starting p-optimal experimental loop \n');
count=0;        %preset to zero
OPTIONS=optimset('Display','iter', 'LargeScale', 'off');
for i=1:ni
        popt{7}=i;              %value of iteration needs to be passed to some of the functions
        iterp=i                 %this is used to determine how many iterations of each exp. design was performed
        best=popt{8}(i);        %stores the number of most probable model for this iteration
        if flag2(2)==1  %flag2 tells whether or not to run box algorithm (1=yes)
                [pointp,XY4,Z4,Z1]=optgraph(popt{4}(1,:),popt{4}(2,:),step1,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i), 4);  %This is used to generate XY and Z for box1.m inputs
                %box1 determines the reduced experimental area based on the CI of est. optimal point
                [pout{i},popt{15}(i,:),popt{9}(i)]=box1({XY4{:}},Z4,Z1,{popt{:}},{presult{:,best}},{model{:,best}},2);
                %gridint.m determines the area around the optimal point for the optimality function calculation
                [LB1,UB1]=gridint(popt{15}(i,:),Low2,Up2,step1);
                [popt{5,1}(i,:),fnew] = fmincon(@(xd)optfun(xd,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i),2,sc2),popt{15}(i,:),[],[],[],[],LB1,UB1);  %finds best experimental point by minimizing optfun.m
    else                        %
                %first, find best starting point for optimization using optgraph
                [pointp2(i,:),XY5, Z5(:,:,i),Z6]=optgraph(popt{4}(1,:),popt{4}(2,:),step1, presult{1,best}(i,:),{model{:,best}}, {popt{:}},{presult{:,best}},presult{11,best}(i),2);
                [popt{5}(i,:),fnew,Gp] = fmincon(@(xd)optfun(xd,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i),2,sc2),pointp2(i,:),[],[],[],[],popt{4}(:,1),popt{4}(:,2),[],OPTIONS);
    end
        popt{17}(:,i)=scale(popt{5}(i,:),1);    %stores descaled values of experimental points
        %generate new data point
        [data5]=Simdata(popt{5}(i,:), rep,err1(z));
        praw=[praw;data5];      %stores raw data
        popt{2}=[popt{2};data5(:,1:3)];         %adds new data to existing data set
    for j=1:mC          %goes through each of 'mC' models
            %calculate metrics for each model
            [presult{1,j}(i+1,:),presult{2,j}(i+1),presult{4,j}(i+1,:),presult{5,j},presult{6,j}(i+1),presult{7,j}(i+1),presult{8,j}(i+1),presult{14,j}(i+1),presult{9,j}(i+1),presult{15,j}(i+1),presult{10,j}(i+1,:),presult{11,j}(i+1),presult{12,j}(i+1),presid{i},popt{1,1}(i+1),presult{16,j}(i+1),presult{18,j}(i+1),presult{19,j}(i+1),presult{20,j}(i+1),pdata{j,i+1},pndata{j,i+1},presult{21,j}(i+1,:),presult{22,j}(i+1),presult{23,j}(i+1)]=metrix3({model{:,j}},{popt{:,1}},{presult{:,j}},step2);
            presult{17,j}(:,i+1)=scale(presult{4,j}(i+1,:),1);  %descales T and C
            allP(j)=mprob(iprob,model{2,j}, popt{6}, presult{2,j}(i+1));     %calculate probabilities for each model
    end
    %calculate normalized probabilities for each model and find most
    %probable model
    [P,popt{8,1}(i+1)]=maxP(allP);
    for j=1:mC
            %assign probabilities to the respective model arrays
            presult{3,j}(i+1)=P(j);
    end
    exper=size(popt{2});       %calculates how many experiments have been run
    [pout1(i+1,:),count,popt{11}(i+1,:)]=SC2(popt{7},{presult{:,popt{8}(i+1)}},goal,count,exper(1)
      ,rep,popt{16});
    if pout1(i+1)==2             %if stopping criterion function SC2.m are met twice in a row, pout
      =2 and simulation should end
            if flag2(4)==1              %simulation ends if flag2(4)=1, which means to stop
                break
            else
                pout1(i+1)=0;           %if SC is not being run, change out to zero so box
                  algorithm can be run
            end
    end
end
%-----------------------------Saving Data------------
        p=datestr(now,'mm_dd_yy');
        ch5=[p,'_',num2str(z),'_',num2str(num)];
        save(ch5);
clear point
count=0;         %preset to zero
%**************************D-optimal*********
fprintf('starting d-optimal experimental loop \n');
        for i=1:ni
                dopt{7}=i;
                iterd=i
                best=dopt{8}(i);         %stores the number of most probable model for this
                  iteration
                if  flag2(2)==1 %flag2 tells whether or not to run box algorithm (1=yes)
                         [pointd,XY4,Z4,Z1]=optgraph(dopt{4}(1,:),dopt{4}(2,:),step1,dresult{1,
                           best}(i,:), {model{:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best
                           }(i), 4); %This is used to generate XY and Z for box1.m inputs
                         [dout{i},dopt{15}(i,:),dopt{9}(i)]=box1({XY4{:}},Z4,Z1,{dopt{:}},{dresult
                           {:,best}},{model{:,best}},1);
                         [LB1,UB1]=gridint(dopt{15}(i,:),Low2,Up2,step1);
                         [dopt{5,1}(i,:),fnew] = fmincon(@(xd)optfun(xd,dresult{1,best}(i,:),{
                           model{:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best}(i),1,sc2),
                           dopt{15}(i,:) ,[],[],[],[],LB1,UB1);
                else                            %run normal optimazation routine
                        %first, find best starting point for optimization using optgraph
                        [pointd2(i,:),XY5, Z5(:,:,i),Z6]=optgraph(dopt{4}(1,:),dopt{4}(2,:),step1,
                           dresult{1,best}(i,:),{model{:,best}}, {dopt{:}},{dresult{:,best}},
                          dresult{11,best}(i),1);
                        [dopt{5}(i,:),fnew] = fmincon(@(xd)optfun(xd,dresult{1,best}(i,:),{model
                          {:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best}(i),1,sc2),pointd2
                          (i,:) ,[],[],[],[],dopt{4}(:,1),dopt{4}(:,2),[],OPTIONS);
                end
                dopt{17}(:,i)=scale(dopt{5}(i,:),1);    %stores descaled values of experimental
                  points
                %generate new data point
                [data5]=Simdata(dopt{5}(i,:), rep,err1(z));
                draw=[draw;data5];
                 dopt{2}=[dopt{2};data5(:,1:3)];
        for j=1:mC
                 %calculate metrics for each model
                [dresult{1,j}(i+1,:),dresult{2,j}(i+1),dresult{4,j}(i+1,:),dresult{5,j},dresult
                  {6,j}(i+1),dresult{7,j}(i+1),dresult{8,j}(i+1),dresult{14,j}(i+1),dresult{9,j}(
                  i+1),dresult{15,j}(i+1),dresult{10,j}(i+1,:),dresult{11,j}(i+1),dresult{12,j}(i
                  +1),dresid{i},dopt{1}(i+1),dresult{16,j}(i+1),dresult{18,j}(i+1),dresult{19,j}(
                  i+1),dresult{20,j}(i+1),ddata{j,i+1},dndata{j,i+1},dresult{21,j}(i+1,:),dresult
                  {22,j}(i+1),dresult{23,j}(i+1)]=metrix3({model{:,j}},{dopt{:,1}},{dresult{:,j
                  }},step2);
                 dresult{17,j}(:,i+1)=scale(dresult{4,j}(i+1,:),1);  %descales T and C
                allP(j)=mprob(iprob,model{2,j}, dopt{6}, dresult{2,j}(i+1));     %calculate
                  probabilities for each model
        end
        %calculate normalized probabilities for each model and find most
         %probable model
        [P,dopt{8,1}(i+1)]=maxP(allP);
        for j=1:mC
                %assign probabilities to the respective model arrays
                dresult{3,j}(i+1)=P(j);
        end
        exper=size(dopt{2});      %calculates how many experiments have been run
        [dout1(i+1,:),count,dopt{11}(i,:)]=SC2(dopt{7},{dresult{:,dopt{8}(i+1)}},goal,count,exper
          (1),rep,dopt{16});
        if dout1(i+1)==2
                if flag2(4)==1
                        break
                else
                        dout1(i+1)=0;           %if SC is not being run, change out to zero so box
                           algorithm can be run
                end
        end
end
%------------------------------Saving Data---------
        p=datestr(now,'mm_dd_yy');
        ch5=[p,'_',num2str(z),'_',num2str(num)];
        save(ch5);
clear point
count=0;        %preset to zero
%**********************RANDOM*****************
fprintf('starting random experimental loop \n');
for i=1:ni
        gopt{7}=i;
        iterg=i
        best=gopt{8}(i);        %stores the number of most probable model for this iteration
        if flag2(2)==1  %flag2 tells whether or not to run box algorithm (1=yes)
                [pointg,XY4,Z4,Z1]=optgraph(gopt{4}(1,:),gopt{4}(2,:),step1,gresult{1,best}(i,:),
                  {model{:,best}},{gopt{:}},{gresult{:,best}},gresult{11,best}(i), 4);  %This is
                  used to generate XY and Z for box1.m inputs
                [gout{i},gopt{15}(i,:),gopt{9}(i)]=box1({XY4{:}},Z4,Z1,{gopt{:}},{gresult{:,best
                  }},{model{:,best}},4);
                [o,q]=size(gout{i});
                if q==8                 %if this is true, then there is at least one point that is
                   within boxmax, and points are chosen randomly from available list of points for
                   next experiment.  If q=1, then an experiment is generated randomly from entire
                  experimental space
                      pop=round(unifrnd(1,o));
                      gopt{5,1}(i,:)=gout{i}(pop,3:4);
        else                    %if points can't be found on the grid, then make random points
            gopt{5,1}(i,1) = unifrnd(gopt{4}(1,1),gopt{4}(1,2));
            gopt{5,1}(i,2)= unifrnd(gopt{4}(2,1),gopt{4}(2,2));
        end
    else
        %generate experimental points using random number generator, unifrnd picks random number between optrange
        gopt{5,1}(i,1) = unifrnd(gopt{4}(1,1),gopt{4}(1,2));
        gopt{5,1}(i,2)= unifrnd(gopt{4}(2,1),gopt{4}(2,2));
    end
    gopt{17}(:,i)=scale(gopt{5}(i,:),1);    %stores descaled values of experimental points
    %generate new data point
    [data5]=Simdata(gopt{5}(i,:), rep,err1(z));
    graw=[graw;data5];
    gopt{2}=[gopt{2};data5(:,1:3)];
    for j=1:mC
        %calculate metrics for each model
        [gresult{1,j}(i+1,:),gresult{2,j}(i+1),gresult{4,j}(i+1,:),gresult{5,j},gresult{6,j}(i+1),gresult{7,j}(i+1),gresult{8,j}(i+1),gresult{14,j}(i+1),gresult{9,j}(i+1),gresult{15,j}(i+1),gresult{10,j}(i+1,:),gresult{11,j}(i+1),gresult{12,j}(i+1),gresid{i},gopt{1,1}(i+1),gresult{16,j}(i+1),gresult{18,j}(i+1),gresult{19,j}(i+1),gresult{20,j}(i+1),gdata{j,i+1},gndata{j,i+1},gresult{21,j}(i+1,:),gresult{22,j}(i+1),gresult{23,j}(i+1)]=metrix3({model{:,j}},{gopt{:,1}},{gresult{:,j}},step2);
         gresult{17,j}(:,i+1)=scale(gresult{4,j}(i+1,:),1);  %descales T and C
        allP(j)=mprob(iprob,model{2,j}, gopt{6}, gresult{2,j}(i+1));     %calculate probabilities for each model
    end
    %calculate normalized probabilities for each model and find most
    %probable model
    [P,gopt{8,1}(i+1)]=maxP(allP);
    for j=1:mC
        %assign probabilities to the respective model arrays
        gresult{3,j}(i+1)=P(j);
    end
    exper=size(gopt{2});      %calculates how many experiments have been run
    [gout1(i+1,:),count,gopt{11}(i+1,:)]=SC2(gopt{7},{gresult{:,gopt{8}(i+1)}},goal,count,exper(1),rep,gopt{16});
    if gout1(i+1,1)==2
        if flag2(4)==1
            break
        else
            gout1(i+1,1)=0;                 %if SC is not being run, change out to zero so box algorithm can be run
        end
    end
end
clear point
finish=datestr(now)     %outputs time when simulation is finished
%---------------------------Saving Data-----------
    p=datestr(now,'mm_dd_yy');
    ch5=[p,'_',num2str(z),'_',num2str(num)];
    save(ch5);

```

```
%% Save all figures
SaveAllFigures(p,z,num);
close all       %this command closes all graphs
end
 end
%% extra functions needed for code to run are all included in the following section
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%box1%%%%%%%%%%%%%%%%%%%%%
function [out,point,boxmax]=box1(mat,Z,Zall,opt,result,model8,flag)
%**********modified for nucleation study********
%this function takes as input the array of x and y values 'X', and the
%matrix of some functions evaluations in the (x,y) space, Z.  'boxmax' is the
%maximum allowed value calculated from avareage output + CI
%Z1 is the model evaluations at (X,Y) where Z is the objective function evaluations at (X,Y)

%this function calculates whether points outputed from optgraph (mat, Z, and Zall) are potential optimal points of the process based on the 'boxmax'
%---------Constraints------------------------------
ntarget=0.001;          %smallest value of n_{isl} desired
ttarget=3;              %want film to grow in 3 minutes
%--------------------------------------------------
a=opt{7};       %iteration
counter=0;      %need something to track how many points are below boxmax
theta=result{1}(a,:);   %parameters
sigsq=result{11}(a);    %exp error
hand=model8{1};         %handel for model function
d=opt{1}(a);            %max value of data for Jac.m
J=result{5};            %X matrix (derivative of model wrt parameters)
n=length(opt{2});       %calculates how many experiments have been performed
delta=result{12}(a);    %confidence interval on F
p=model8{2};            %# parameters in model
pred=result{9}(a);      %flux prediction from model
t=result{22}(a);                %value of t from model at est. opt. point
nisl=result{18}(a);
value=result{16}(a);    %value of obj. func. at est. opt. point of most probable model
%Calculation of variance on objective function
CI=result{21}(a,3);     %confidence interval calculated in metrix3 for optimal point
boxmax=value+CI;        %
X=mat{1};       Y=mat{2};               %X and Y matrices generated from optgraph.m
Z1=Zall{1};     Z2=Zall{2};     Z3=Zall{3};     Z4=Zall{4};     %taking Zall apart to be used by box.m function (Z1=time, Z2=nisl, Z3=F, Z4=dnisl)
for i=1:length(X)
        for j=1:length(X)
                point=[X(i,j) Y(i,j)];
                %calculate prediction variance at point
                if model8{6}==0, Jopt=Jac(theta,hand,[X(i,j) Y(i,j)],d);
                elseif model8{6}==1 && model8{7}==1, Jopt=ones(1,1);
                else  Jopt=[ones(1,1) X(i,j) Y(i,j)];  end
                PVopt1=PV(J,Jopt,sigsq);
                %calculate confidence interval at estimated optimal point.  Equation taken
                %from pg 395 Montgomery.  **Jopt should be a 3x1 matrix, so the transpose is
                %reversed**
                delta2=tinv(0.975,(n(1)-p))*sqrt(sigsq*(Jopt*inv(J'*J)*Jopt'));
                CI2=abs(-150*(Z3(i,j)*0.2651)^(-2))*delta2;             %error propagation for time calculation
```

```
                    CI3=abs(Z4(i,j))*delta2;                                %error propagation for
                      n_isl calculation
                    CI4=sqrt(abs(2*(Z1(i,j)-ttarget)))*CI2+sqrt(abs(2*(1e4*(Z2(i,j)-ntarget))))*CI3;
                            %calculates the uncertainty in each point
                    if (Z(i,j) < boxmax) || ((Z(i,j)-CI4) < boxmax)    %want value to be greater than
                      boxmax since the point is to maximize N_isl
                            counter=counter+1;
                            out(counter,:)=[Z1(i,j) CI4 X(i,j) Y(i,j) Z(i,j) Z2(i,j) CI boxmax];
                              %the column indice of Z is the x indice, row is for y indice
                    end
            end
end
if counter==0            %if nothing was found that met contstraints, end function
    out=1;               %out returns a dummy variable so no error
    return
end
%out is of the form [output x y] for as many rows as values below 'boxmax'
% part 2 of function----picking best value for pfun
[n,m]=size(out);                  %calculates how many points were found
if flag==4               %if flag==4, doing random and want to skip this set of calculations
        point=[out(1,3) out(1,4)];      %if flag==4, this point is meaningless
else
        for k=1:counter        %calc value of optimality (depends on flag) at the points below
          Pvalue
                out(k,9)=optfun([out(k,3) out(k,4)],theta,model8,opt,result,sigsq,flag);
        end
        %----------FINDING MINIMUM--------------
        [val,I]=min(out(:,9));          %finds minimum of each column in 'b' and its indice
        xfin=out(I,3);
        yfin=out(I,4);
        point=[xfin yfin];        %index of minimum point
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Flux%%%%%%%%%%%%%%%%
function [F,t]=flux(u,err)
%This function outputs flux [=]ML/min
%this function assumes 300 ML/min is maximum flux possible
%highest T is 1073K (700C)
% gas flow=20 sccm
%C is concentration and should be [=] micromol/L
%T is temperature and should be [=]K
%------------------constants used---------------
R=0.00831447;              %kJ/(mol*K)
Avo=6.022e23;              %Avogadro's number  [=]atoms/mol
flow=5.061;                %[=]L/min = 20 sccm  (taken from Calculations folder)
MW=88;                     %molecular weight of Y [=]g/mol
D=2.54;                    %diameter of circular substrate [=]cm
conv=1e7;                  %conversion [=]nm/cm
av=0.0745;                 %atomic volume  [=]nm^3/atom
Ea=19.25;                  %activation energy for flux  (calculation on pg 47 of notebook)
conv2=0.2651;              %conversion for nm to monolayers (ML) [=]nm/ML
%--------------Area of wafer calc----------------
Area=pi*(D*conv/2)^2;   %assumes circular substrate
%descaling variables
z=scale(u,1);
```

```
T = z(1); %temperature [K]
C = z(2); %Concentration [micromoles/L]
%---------------Fo calculation-----------------
Fo=C*1e-6*flow*Avo*av/Area/conv2;
%---------------Flux calculation---------------
F=Fo*exp(-Ea/(R*T))+normrnd(0,err);
t=150/(F*conv2);          %target of 150nm thick film
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%metrix3%%%%%%%%
function [theta,Sr,point,J,Dopt,PVx1,PVopt1,err1,pred,pred2,F,sigmasq,CI,r,d,val,nisl,truen,
    counter,data1,ndata1,CI5,tm,dnisl]=metrix3(model2,opt,result,step)
%**********Modified for nucleation simulation*********
%This function will reduce the clutter in the main function and does calculations that would've
    been repetitive anyway.
%%%%INPUT Definitions
%model2=array passed with needed information for optimality
%err=experimental error
%range=lower and upper bounds for calculating new thetas
%optrange=range for finding optimum point
%step=how many steps one wants optgraph to perform
%constants for MHF
%%%%OUTPUT Definitions
%theta=predicted thetahat for this iteration
%Sr=predicted Srhat for this iteration
%d=predicted max value of y from data
%J=predicted jacobian for this iteration
%data=data matrix plus new experiment from this iteration
%Dopt=D-optimal value for this iteration
%PVx1=prediction variance for experimental point
%pred=predicted value at estimated optimal point
%err1=error of predicted value to actual value at predicted optimal point
%PVopt1=PV at initial estimated optimal point
%PVopt2=PV at real optimal point
%pred2=output from system
%F= F statistical values for LOF (1) is for model (2) is F for comparison
    %to (1)
%sigmasq=experimental variance calculated from experimental data
%optpoint=estimated optimal point for this iteration (after new experiment
%is generated (used in calculations for next round
%------------pulling information from arrays
a=opt{7};                 %passes # of iteration to find correct value
data=opt{2};              %experimental data
hand=model2{1};           %model function handle
exppt=opt{5}(a,:);        %next experimental point
xLB=opt{4}(:,1);          %range for x1
xUB=opt{4}(:,2);          %range for x2
p=model2{2};              % # parameters in model
sc=opt{14};          %scaling for the error
rep=opt{6};          %number repititions
%--------Constraints---------------------------
ntarget=0.001;            %smallest value of n_{isl} desired
ttarget=3;                %want film to grow in 3 minutes
ndata1=data;
%calculate new thetahat and Srhat
counter=0;
```

```
zoink=0;
ndata2=data(1,:);                %initialized to 1, ndata2 used to check if ndata1 has changed from
    one iteration to the next
while zoink==0 && counter < 5                     %only want to run pinky five times to make sure no
    infinite loop.
        %all these calculations are necessary to run before pinky
        hand
        [theta, Sr, G] = newfit(ndata1, model2,opt{13},sc);
        hand
        n=size(ndata1);
        sigmasq=Sr/(n(1)-p);              %calculating experimental variance
        d=max(round(ndata1(:,3)*100))/100;                %calculates maximum of output for
          jacobian calculation
        %*1000 is used so d is a number other than 0
        if model2{6}==0,
                J=Jac(theta,hand,ndata1,d);   %calculates Jacobian of ALL the data to use in D-
                  optimal.
        elseif model2{6}==1 && model2{7}==1, J=ones(n,1);
        else J=[ones(n,1) ndata1(:,1) ndata1(:,2)];
        end
        hand
        %find optimal point predicted by the optimality
        %first, use optgraph to start optimization
        [pointp,XY4,Z4,Z1]=optgraph(opt{4}(1,:),opt{4}(2,:),step, theta, {model2{:}},{opt{:}},{
          result{:}},sigmasq, 4);
        hand
        OPTIONS = optimset( 'Display', 'iter', 'LargeScale', 'off');
        [point,val]=fmincon(@(point)objfun(point,theta,hand,d),pointp,[],[],[],[],xLB,xUB);
        [val,stuff,dnisl]=objfun(point,theta,hand,d);   %need to recalc objfun.m now that opt.
          point is found for other parameters needed for error propagation
        tm=stuff(2);     nisl=stuff(3);  pred=stuff(1);
        hand
        if model2{6}==0, Jopt=Jac(theta,hand,point,d);
        elseif model2{6}==1 && model2{7}==1, Jopt=ones(1,1);
        else  Jopt=[ones(1,1) point(1) point(2)];  end
        %calculate confidence interval at estimated optimal point. Equation taken
        %from pg 395 Montgomery. **Jopt should be a 3x1 matrix, so the transpose is
        %reversed**
        CI=tinv(0.975,(n(1)-p))*sqrt(sigmasq*(Jopt*inv(J'*J)*Jopt'));
        %Calculate prediction at estimated optimal point
        CI2=abs(-150*(pred*0.2651)^(-2))*CI;            %error propagation for time calculation
        CI3=abs(dnisl)*CI;                              %error propagation for n_isl calculation
        CI4=sqrt(2*abs(tm-ttarget))*CI2+sqrt(2*abs(1e4*(nisl-ntarget)))*CI3;    %include both
          terms in objective function since both depend on F
        CI5=[CI2 CI3 CI4];               %array to store all confidence intervals
        boxmax=val+CI4; %
        [ndata1,data1]=pinky(data,boxmax,theta, hand,p,d);      %this function will determine if
          data used to fit the parameters is within confidence interval.  If it is not, that point
           is removed and the parameters are refit using revised experimental data
        counter=counter+1;               %increment counter, only want to do three iterations
          through this loop
    n5=size(ndata1); n6=size(data); n7=size(ndata2);
        if (n5(1)==n6(1)) || (n5(1)==n7(1) && ndata1(1,3)==ndata2(1,3)) %if sizes are equal,
          either it's already using all the data, or it is unchanged from one iteration to the
          next, meaning we should get out of this while loop
```

```
                zoink=1;
        end
        ndata2=ndata1;
end
%Prediction Variance at new point
if model2{6}==0, Jnew=Jac(theta,hand,exppt,d);
elseif model2{6}==1 && model2{7}==1, Jnew=ones(1,1);
else  Jnew=[ones(1,1) exppt(1) exppt(2)];  end
PVx1=PV(J,Jnew,sigmasq);
%calculate prediction variance at estimated optimal point
[dummy1,dummy2,dummy3]=Simdata(point,1,0,1);
pred2=dummy1(3);        %flux of deposition from true system
truen=dummy3(1);        %nucleation density from true system
err1=(pred-pred2)^2;                %error calculation at estimated optimal
                                     %point
PVopt1=PV(J,Jopt,sigmasq);
%calculate prediction variance at real optimal point (not using this since we don't know optimal
    point for nuc simulation)
%calculate final D-optimal value using function 'dfun'
Dopt=optfun(exppt,theta, model2,opt,result,sigmasq,3);
F=50;%LOF(Sr,data(:,3),rep,p);
%calculate studentized residuals
[Sr2,resid]=newerr(theta,data1,hand,sc);  %finds residuals from newerr
Hat=J*(J'*J)^(-1)*J';   %calculates hat matrix which relates error in residuals to experimental
    errors
for i=1:n(1)
   r(i)=resid(i)/sqrt(Sr*(1-Hat(i,i)));
end
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%newfit%%%%%%%%%%%%%%%%%%%%%%%
function [theta,Srsc,G] = newfit(data,model,sett,sc)
%this function fits the parameters to the data by minimizing the error
%**********modified for use with nucleation modeling*****
%function calculates the best fit parameters to the data , theta, as well as the
%model error, Sr
% data in [y x] form
%model is the function handle
%po is the parameter guess
%range contains the bounds for the optimization
%Sr0 is initial error
%theta, Sr, G, output are outputs of fmincon
%------------------------------------------------------
%if statement was added with mod_avg in mind.  mod_avg needs its new
%parameters to just be an average of the available data, which won't
%necessarily minimize the error
%------------------------------------------------------
po=model{3};            %initial parameter guess
hand=model{1};          %model function handle
if model{7}==1
    theta=mean(data(:,3));
    Sr = newerr(theta,data,hand,1);
    G=1;
else
Sr=0; Sr0=0;
```

```
[Sr0,stuff] = newerr(po,data,hand,1); % Initial error in prediction
% Minimize over the fit
%setting tolerance for fmincon
OPTIONS = optimset( 'Display', 'iter','MaxFunEvals', sett(2,1), 'LargeScale', 'off');
%  'TolFun',sett(1,1), 'TolX',sett(1,2),'TolCon',sett(1,2),            --this could go back in
    later
[theta,Sr,G,output] = fmincon(@(theta)newerr(theta,data,hand,1),po,...
                         [],[],[],[],model{4},model{5},[],OPTIONS);
%G is exitflag of fmincon
%output gives information on optimization
%G, output used for debugging
end
Srsc=Sr/1;       %descales Sr to use in rest of simulation
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%nuc%%%%%%%%%%%%%%%%%%
function dndt = nuc(t,n,x,q)
% Nucleation model from Evans, Thiel, and Bartelt
%This is to go with function 'simdata.m' and objfun.m
% Process inputs (factors)
T = x(1);
F = x(2);
%q are the fitted parameters in this model
%for mod_mech2, 'q' has length 2 and 'Ed' must be set to 0.8
if length(q)==3
    Ei = q(1); % Energy of critical cluster [eV]
    Ed = q(2); % diffusion activation energy [eV]
    sigma = q(3);   % Capture number
else
    Ei = q(1); % Energy of critical cluster [eV]
    sigma = q(2);   % Capture number
    %6/26---testing to see if Ed will make a difference in model if it's wrong.
    %Normally at 0.8
    Ed = 0.8; % diffusion activation energy [eV]
end
%constants
kb = 8.62e-5;  % Boltzmann's constant [eV/K]
v = 1e13; %attempt frequency of hopping [1/s]
ci = 1; %# configurations of a stable island
i=2;          %#adatoms needed to form an island
 %(set at low numbers, 1 or 2)
beta =1/kb/T; %[1/eV]
h = v*exp(-beta*Ed);  %[1/s]
%calculation of N_isl
% States
theta = n(1);
N1 = n(2);
Nisl = n(3);
Ni = ci*exp(-beta*Ei)*N1^i; % Density of critical clusters
Knuc = sigma*h*N1*Ni;
Kagg = sigma*h*N1*Nisl;
% Differential equation model
dndt = zeros(3,1);
dndt(1) = F;   % Coverage theta
dndt(2) = F*(1-theta) - (i+1)*Knuc - Kagg;
dndt(3) = sigma*h*N1*Ni;
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%objfun%%%%%%%%%%%%%%%%%
function [z1,r,deriv]=objfun(x,theta,hand,d)
%this function calculates the value of the objective function for the
%nucleation study
%--------Constraints---------------------------
ntarget=0.001;          %smallest value of n_{isl} desired
ttarget=3;              %want film to grow in 3 minutes
%if you change these constraints, must also change constraints in box1.m
%------Calculation ---------------------------
[F,t, nisl]=feval(hand,x,theta);         %gets model prediction of n_{isl}
%z=*10000;              %if nisl < nopt, then z is negative.  10000 is to make values closer to 1
z1=((t-ttarget)^2+(1e4*(nisl-ntarget))^2);                          %objective function for maximizing
    n_isl and minimizing growth time.  Negative sign is used so fmincon finds the point where the
   objective function is at its MAXIMUM
%5e9/(ti-15)^2+
if nargout >=2, r=[F t nisl];, end
if nargout==3, %need to calculate derivative of N_isl calc for error propagation calculation
        dsig=0.001*d;
        dF=F+dsig;
        z=scale(x,1);
        T = z(1); %temperature [K]
        u=[T;F];
        %----------PARAMETERS%-------------------------
        Ei = 0; %binding energy [eV]
        Ed = 0.8; % diffusion activation energy [eV]
        sigma = 0.2;  % capture #
        q=[Ei Ed sigma];        %stores Ei,E_d, and sigma in array to pass to nuc
        n0 = [0; 0; 0];  % Start with a bare Si surface
        thetaf = 0.15;  % Final coverage in monolayers
        tf = thetaf/dF;    % Final time to reach thetaf
        dt = tf/100;    %size of time step
        trange = [0:dt:tf];  % Range of times [s]
        [teh,dn] = ode23t(@nuc,trange,n0,[],u,q);
        deriv=(dn(end,3)-nisl)/dsig;
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%optfun%%%%%%%%%%%%%%%%%
function z=optfun(xd,theta, model4,opt,result,sigmasq2,flag,sc)
%*********modified for use with nucleation model******
%%%This is a function for optimal design.  Depending on the number of
%inputs, the function will do either d-, g-, or p- optimal design.
%-----------------List of variables-------------
%model4=handle for model function
%opt=cell array for optimality being run
%flag=tells function to run either
    %0 = gfun
    %1=dfun
    %2=pfun
%result is cell array of simulation results
hand=model4{1};
a=opt{7};               %passes iteration so it can be found in an array
%first, need to find jacobian for new experimental point
if model4{6}==0, J1=Jac(theta,hand,xd,opt{1}(a));
elseif model4{6}==1 && model4{7}==1, J1=ones(1,1);
```

```matlab
else  J1=[ones(1,1) xd(1) xd(2)];  end
X=[result{5}; J1];

if flag ==0
    %do gfun-tries to reduce PV over entire experimental range by picking
    %point with maximum PV and running next experiment at that point
    z = -J1*inv(result{5}'*result{5})*J1'*sigmasq2;

elseif flag ==1
    %calculates D-optimality (maximizes det(X'*X) )
    z = -det(X'*X);
elseif flag==2
   %do pfun -pfun uses optimal point and experimental point to find the
   %prediction variance at the optimal point.  When used with fmincon, it
   %will find the experimental point with the largest prediction variance
   %and run more experiments at that point
   xp=result{4}(a,:);      %optimal point
    %compute jacobian for optimal point 'xp'
    if model4{6}==0, Jnew=Jac(theta,hand,xp,opt{1}(a));
    elseif model4{6}==1 && model4{7}==1, Jnew=ones(1,1);
    else  Jnew=[ones(1,1) xp(1) xp(2)];  end
    %compute prediction variance
    z = -Jnew*inv(X'*X)*Jnew'*sigmasq2;
elseif flag ==3
    %calculates D-optimality for metrix3 (maximizes det(X'*X) )
    z = -det(result{5}'*result{5});
end
if nargin ==8, z=z*sc;  end     %this is included since the magnitude of optfun for nucleation study is so small, need scaling factor for fmincon
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%PINKY%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [ndata1,ndata]=pinky(data,boxmax,theta, hand,p,d)
%This function is designed to check whether exp. data points fall within confidence interval of desired objective function.  If they don't, remove points from data matrix while still leaving matrix full rank

n=size(data);           %determines how many points are in ndata
%first, need to calculate objective function for each of the data points
for i=1:n
        a=[data(i,1) data(i,2)];
        [z1(i,:),r(i,:)]=objfun(a,theta,hand,d);
end
ndata=[data z1 r];      %stores experimental data in ndata array for editing
ndata=sortrows(ndata,4);
counter=0;
for i=1:n
        if ndata(i,4) < boxmax
                counter=counter+1;      %if yes, increment counter
                ndata1(counter,:)=ndata(i,1:3);
        end
end
if counter==0
    ndata1=ndata(1,1:3);
end
n1=size(ndata1);
```

```
zoink=0;
while zoink==0
        J=Jac(theta,hand,ndata1,d);
        cond=rank(J);   %only need data matrix to be full rank, not experimental output (column 3)
        n1=size(ndata1);
        if (cond ~= p) || (n1(1) <= p)
                ndata1=[ndata1;ndata(n1(1)+1,1:3)];
        else
                zoink=1;                %change flag to get out of while loop
        end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Scale%%%%%%%%%%%%%%%%%%
function z=scale(x,flag)
%should output z as a column vector [T;C]
%flag =1 means descale, otherwise, scale variables
Ta=973;         Tb=100;
Ca=0.9; Cb=0.6;
if flag==1
        %this function descales the x values so they can be expressed in K and micromoles/min
           respectively
        z(1,1)=x(1)*Tb+Ta;              %Temp
        z(2,1)=x(2)*Cb+Ca;                      %concentration
else    %Scale variables
        %this function scales the x values so they can be expressed in K and micromoles/min
           respectively
        z(1,1)=(x(1)-Ta)/Tb;            %Temp
        z(2,1)=(x(2)-Ca)/Cb;                    %concentration
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Simdata%%%%%%%%%%%%%%%%%%
function [data, q,data2]=Simdata(u,rep,err,flag1)
% Evans, Thiel, and Bartlett (2006) equation (6.4)
%This is the equation used to generate the data from experimental runs
%suggested by D(x)
%INPUTS
%u(1)=T   (scaled)
%u(2)=C   (scaled)
%z=# for saving output graph
%rep=# of repititions to be done
%err=variance of gaussian data
if nargin==2, err=0; end        %if no error is specified, make the error 0
%---------------Un-Scaling variables-------------
z=scale(u,1);
T = z(1); %temperature [K]
[F,t]=flux(u,err);              %calculates flux using 'flux.m'
x = [T; F];    % Temperature and flux
data=[u(1) u(2) F t];
if nargin==4
        %-----------PARAMETERS%-------------------
        Ei = 0; %binding energy [eV]
        Ed = 0.8; % diffusion activation energy [eV]
        sigma = 0.2;  % capture #
        q=[Ei Ed sigma];       %stores Ei,E_d, and sigma in array to pass to nuc
        n0 = [0; 0; 0];  % Start with a bare Si surface
        thetaf = 0.15;  % Final coverage in monolayers
        tf = thetaf/F;    % Final time to reach thetaf
        dt = tf/100;    %size of time step
        trange = [0:dt:tf];  % Range of times [s]
        %This while loop will do 10 repititions of the experiment, save the desired
        %data, and plot the 10th simulation and save it for later viewing
        L = 1;
        while L <= rep;
                [t,n] = ode23t(@nuc,trange,n0,[],x,q);
                data2(L,:)=[n(end,3) tf max(n(:,2))];
                L = L + 1;
        end
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%mod_emp%%%%%%%%%%%%%%%
function [F,t,N_isl]=mod_emp(x,q)
%This is an empirical model for the data from the nucleation modeling
%T=temperature [K]
%i= # adatoms needed to form an island [unitless]
%N_isl=island density
%C=concentration [micromol/L]
%Tu and Cu are scaled variables, need to be unscaled for 'flux.m'
conv2=0.2651;           %conversion for nm to monolayers (ML) [=]nm/ML
%z=scale(x,1);      %empirical will use unscaled variables at this point
%4/15/08
T = x(1); %temperature [K]
C = x(2); %Concentration [micromoles/L]
a=q(1);
b=q(2);
c=q(3);
F=a+b*T+c*C;
t=150/(F*conv2);        %target of 150nm thick film
if nargout==3
        [dummy,dummy2,n]=Simdata(x,1,0,1);      %if there are three outputs to this function, then
           the nucleation density is calculated
    N_isl=n(1);
end
end
%%%%%%%%%%%%%%%%%%%%%%%%mod_avg%%%%%%%%%%%%%%%%%%
function [F,t,N_isl]=mod_avg(x,q)
%This is a model for the data from the nucleation modeling that just takes
%the average of the data and returns that as N_isl
%T=temperature [K]
%N_isl=island density
%F=flux to surface
%These are the dependant variables
%(need to be descaled for mechanistic model)
conv2=0.2651;           %conversion for nm to monolayers (ML) [=]nm/ML
%z=scale(x,1);
%T = z(1); %temperature [K]
F=q;
t=150/(F*conv2);        %target of 150nm thick film
if nargout==3
        [dummy,dummy2,n]=Simdata(x,1,0,1);%if there are three outputs to this function, then the
          nucleation density is calculated
    N_isl=n(1);
end
end
%%%%%%%%%%%%%%%%%%%%%%mod_mech%%%%%%%%%%%%%%%%%%%%%%
function [F,tf2,N_isl]=mod_mech(x,q,flag1)
%this function uses a hybrid model to calculate the flux based on activation energy
%%-----------------------------------------
R=0.00831447;            %kJ/(mol*K)
conv2=0.2651;            %conversion for nm to monolayers (ML) [=]nm/ML
%%----------------------------------------
%These are the dependant variables
%(need to be descaled for mechanistic model)
z=scale(x,1);
T = z(1); %temperature [K]
C = z(2); %Concentration [micromoles/L]
%---------------Fo calculation---------------
%---------------Flux calculation-------------
F=Fo*exp(-q/(R*T));
tf2=150/(F*conv2);       %target of 150nm thick film
x1=[T F];
%%%------------------------------------------
if nargout==3
        [dummy,dummy2,n]=Simdata(x,1,0,1);%if there are three outputs to this function, then the
          nucleation density is calculated
    N_isl=n(1);
end
end
%%%%%%%%%%%%%%%%%%%%%%mod_mech2%%%%%%%%%%%%%%%%%%%%%
function [F,tf2,N_isl]=mod_mech2(x,q)
%this function is an empirical model
%-------------------------------------------------
conv2=0.2651;            %conversion for nm to monolayers (ML) [=]nm/ML
%These are the dependant variables
%(need to be descaled for mechanistic model)
z=scale(x,1);
T = z(1); %temperature [K]
C = z(2); %Concentration [micromoles/L]
%---------------Fo calculation---------------
F=C*T*q(1)+q(2);
tf2=150/(F*conv2);       %target of 150nm thick film
if nargout==3
        [dummy,dummy2,n]=Simdata(x,1,0,1);%if there are three outputs to this function, then the
          nucleation density is calculated
    N_isl=n(1);
end
end
```

Listing D.2: Dsim29.m in Matlab

D.3 Matlab code for experimental study

This is the file used to determine the sequential experiments in the experimental study with the CVD reactor testbed.

```
function ni=exp7(nameout,flag2,rep,sctol,num,data)
%7/30/2008      This is modified from 'exp4.m', growth time has been found to be unrepeatable, so
   that portion is being taken out and working solely with roughness
%includes conversion of molar flow to flux for 'nislcalc'
%This is a function to run the simulation code, and takes the names of the
%output file and number of repititions.  Also incorporates F(T,C)
%*******List of inputs and outputs*******************
%nameout= name of output file for reports
%err= desired level of noise to be added to system
%flag2= vector of length 4 for each optimality, tells whether or not to run box algorithm [P D G
   Run] 'Run' is a flag to say whether to run D and G or not, 1=yes
%rep= # repititions to perform for each simulated experiment
%num= # to help differentiate between different simulations
%Srnorm= value of normal Sr (kind of a dummy output)
%List of variables*********************************
%a, avg, b,cells, ch, ch1, ch2, ch3, ch4, col1, col2, count,d, data, deriv, discr, emp, err,
%err1, exp, F, flag, flag1, flag2,FVAL, G, guess,h,i, I, I2, iter, ik,j,k, K, kk,L, Low1, m, mC,
   mech, mod, model,
%n, namein, nameout, nC, ni, num, Options, other,output, p, pM, pMcond, psum, P,
%q, Q, r, R, raw, rep, s,sc, simdata, Srhat, SrLS, SrLSnorm
%scal, scal2, set, sett, step, step2,thetahat, T,tol, u, Up1, v, ve, x, XO, Xset, Xsetnew, z

%**************************************************
%Check Validity of inputs
if ischar(nameout)==0
    error('nameout needs to be a character string, use single quotes')
end
%———————————————————USER INPUTS———————————————
source='exp7';          %Keeping track of what generator file generated what data
goal=5;             %defines desired magnitude of confidence interval (1 second)
flag=0;                 %flag==1 enables graphs at end
ni=6;               %max # of iterations
Low1=[873; 100];  %lower bounds for experimental settings
Low2=scale(Low1,0);     %scales lower bounds
Up1=[1048; 200];  %upper bounds for experimental settings
Up2=scale(Up1,0);       %scales upper bounds
step1=15;                       %number of steps to take when making grid of design space for 1st
   part of discrim function
step2=20;                     %finding optimal point, needed more spots on grid
cells=[24 17];        %defines size of cell arrays for opt and model array
XO=[971 150]; %XO=initial experimental settings for fmincon discrim func;
Xset(1,:)=scale(XO,0);  %scales XO (set of experimental settings, [T C]
%err=1e-4;                 %variance of gaussian noise added to data during simulation
tol=1e-18;            %Tolerance for fmincon
tol2=1e-10;            %Tolerance for TolX
iterate=100;            %iterations for fmincon
sett=[tol tol2;iterate 0];         %puts tolerances and scaling in a vector for fmincon
sc2=1%e17;       %this is a scaling factor for fmincon search on optfun
Pvalue=zeros(ni,1);
```

```
rand('state',sum(100*clock));          %need to initialize seed for random number generator
%------------------global variables--------------------
global R global_conv2 global_thick Ei Ed sigma thetaf kb global_v ci global_i ntarget  conf2
R=0.00831447;            %kJ/(mol*K)
global_conv2=0.2651;            %conversion for nm to monolayers (ML) [=]nm/ML
global_thick=120;               %desired thickness of film [nm]
Ei = 0; %binding energy [eV]
Ed = 0.8; % diffusion activation energy [eV]
sigma = 0.2;  % capture #
thetaf = 0.15;  % Final coverage in monolayers
kb = 8.62e-5;  % Boltzmann's constant [eV/K]
global_v = 1e13; %attempt frequency of hopping [1/s]
ci = 1; %# configurations of a stable island
global_i=2;         %#adatoms needed to form an island
ntarget=7;               %desired roughness [nm]
conf2=0.975;             %level of confidence for confidence intervals
%-----------------------Make model arrays--------------
%this makes the model arrays with information about each model
%models are organized by column
% model(1,:)=name of model
% model(2,:)=%# of parameters in the model
% model(3,:)=model{3,1}=[.7 .1 .7];              %initial guess for emp model parameters
        %initial guess for model parameters
% model(4,:)=Lower bound for model parameters
% model(5,:)=Upper bound for model parameters
% model{6,:}=Flag (==0, no derivative  ==1 derivative in model{8,:}
% model{7,:}=Flag for Compfit (==0, normal   ==1, for mod_avg)
                % available data points
%%************************%Empirical model**********
model{1,1}=@rms1;               %name of model to be loaded
model{3,1}=[0.7 0.1];                  %initial guess for model parameters
model{2,1}=length(model{3,1});                       %# of parameters in the model
model{4,1}=-1000*ones(1,model{2,1});          %Lower bound for emp model
model{5,1}=1000*ones(1,model{2,1});            %Upper bound for emp model
model{6,1}=0;
model{7,1}=0;
%%************************%Empirical model**********
model{1,2}=@rms2;               %name of model to be loaded
model{3,2}=[0.7 0.1 0.3];                  %initial guess for model parameters
model{2,2}=length(model{3,2});                       %# of parameters in the model
model{4,2}=-1000*ones(1,model{2,2});          %Lower bound for emp model
model{5,2}=1000*ones(1,model{2,2});            %Upper bound for emp model
model{6,2}=0;
model{7,2}=0;
%%************************%Empirical model**********
model{1,3}=@rms3;               %name of model to be loaded
model{3,3}=[0.7 0.1];                  %initial guess for model parameters
model{2,3}=length(model{3,3});                       %# of parameters in the model
model{4,3}=-1000*ones(1,model{2,3});          %Lower bound for emp model
model{5,3}=1000*ones(1,model{2,3});            %Upper bound for emp model
model{6,3}=0;
model{7,3}=1;
%%************************%Empirical model**********
model{1,4}=@rms4;               %name of model to be loaded
model{3,4}=[0.7 0.1 0.3];                  %initial guess for model parameters
```

```
model{2,4}=length(model{3,4});                         %# of parameters in the model
model{4,4}=-1000*ones(1,model{2,4});            %Lower bound for emp model
model{5,4}=1000*ones(1,model{2,4});              %Upper bound for emp model
model{6,4}=0;
model{7,4}=0;
%%*************************%Empirical model**********
model{1,5}=@rms5;                  %name of model to be loaded
model{3,5}=[0.7 0.1];                    %initial guess for model parameters
model{2,5}=length(model{3,5});                         %# of parameters in the model
model{4,5}=-1000*ones(1,model{2,5});            %Lower bound for emp model
model{5,5}=1000*ones(1,model{2,5});              %Upper bound for emp model
model{6,5}=0;
model{7,5}=0;
%%*************************%Empirical model**********
model{1,6}=@rms6;                  %name of model to be loaded
model{3,6}=[0.7];                     %initial guess for model parameters
model{2,6}=length(model{3,6});                         %# of parameters in the model
model{4,6}=-1000*ones(1,model{2,6});            %Lower bound for emp model
model{5,6}=1000*ones(1,model{2,6});              %Upper bound for emp model
model{6,6}=0;
model{7,6}=0;
%%*************************Mechanistic model*********
 model{1,7}=@rms7;                  %name of model to be loaded
 model{3,7}=0.7;                    %initial guess for model parameters
 model{2,7}=length(model{3,7});                         %# of parameters in the model
 model{4,7}=-1000;                  %Lower bound for model
 model{5,7}=1000;               %Upper bound for model
 model{6,7}=0;
 model{7,7}=1;
[m,mC]=size(model);            % mC is the number of models contained in models.mat
%-----------------------make opt arrays---------------
%opt cell arrays store info as follows:
%{1}=d                        max value of experimental data
%{2}=data                    stores the simulated data
%{3}=err                      stores error
%{4}= optrange                range for experimental points (upper and lower
        %bounds)
%{5}=experimental point      next experimental point to be run
%{6}= # repetitions to be run
%{7}= element to pass value of 'a' to other functions
%{8} stores number of correct model for each iteration
%{9} stores max value acceptable for box calc
%{10}=system output for stopping criterion
%{11}=verdict                    tells user why simulation was stopped
                            %1=theta and Srhat not changing
                            %2=CI < sqrt(noise)
                            %3=CI < CI (desired)
%{12}=counter                 counter for stopping criterion , see SC for all
                             %stopping criterion questions
%{13}=sett                %this has the settings for fmincon
%{14}=scal                %this carries the scalings to be used for newfit.m and newerr.m
%{15}= stores the points found using box1 and values of function at those points [x1 x2]
%{16}= tolerance for SC (may not be used in growth time simulations..)
%{17}= experimental points descaled
init=feval(@cell,cells(2),1);     %extra cell array for inital calcs
```

```
%————————————————————————result arrays——————————————
%result arrays for each model and optimality
%1=thetahat
%2=srhat
%3=normalized probability of model
%4=estimated optimal point
%{5}=J                      Jacobian of data with respect to parameters
%{6}=D                      D−optimal value of model
%{7}=PVxnew                 prediction variance at new experimental point
%{8}=PVxopt                 prediction variance at estimated optimum point
%{9}=model prediction at estimated optimal point
%{10} holds F statistics for LOF when it is calculated
%{11}=experimental error (calculated)
%{12}=confidence interval on predictions
%{13}=empty
%{14}=model prediction error
%{15}= output from system at optimal point
%{16}=objective function output
%{17}=unscaled estimated optimal point
%{18}=n_isl from model
%{19}=n_isl from true system
%{20}=counter for pinky.m
%{21}=array of confidence intervals from metrix3
%{22}=time needed for film to grow
%{23}=J matrix for optimal point
%{24}=J matrix for optimal point for time model
result=feval(@cell,cells(1),mC);     %extra cell array for inital calcs
%**************************************************
%%———————————————————MAIN CODE———————————————————
%**************************************************
%first find most probable model to work with the growth rate model
allP=zeros(mC,1);               %initialize array to zero
iprob=1/mC;                     %initial probability for models
 for z=1;               %run experiment different times with a different initial parameter guess
%——————————————initial calculations for roughness models
                d=max(round(data(:,3)*100))/100;              %calculates maximum of output for
                    jacobian calculation
        [n,nC]=size(data);
        for nt=1:n
                [dnisl1(nt),extranisldat]=nislcalc(data(nt,1:2));
                %calculates nucleation density for the experimental data
        end
        %preset the opt cell arrays to the correct initial values
        init{2}=data; init{1}=d; init{4}=[Low2 Up2]; init{3}=dnisl1;    init{5}=Xset; init{6}=rep;
            init{7}=1;  init{11}=zeros(1,3);     init{13}=sett;
        init{14}=sc;    init{16}=sctol;
 %——————————%PARAMETER FIT AND PROBABILITY CALCULATION———
            % Compute the least squares fit parameters (theta1hat)and
                    %error for parameters (Sr1hat) for each model
                    %last number in input for compfit cell array for model
            for j=1:mC
                    [result{1,j}(1,:),result{2,j}(1),result{4,j}(1,:),result{5,j},result{8,j}(1),
                      result{11,j}(1),result{12,j}(1),init{1}(1),result{16,j}(1),result{18,j}(1),
                      result{21,j}(1,:),result{23,j}]=metrix3({model{:,j}},{init{:,1}},{result{:,j
                      }},step2);  %
```

```
                        result{17,j}(:,1)=scale(result{4,j}(1,:),1);        %descales T and C
                        model{3,j}=result{1,j}(1,:);          %replaces initial guess with new theta for next iteration
                        allP(j) = mprob(iprob,model{2,j},init{6},result{2,j});  % Compute the conditional probabilities of the models
                end
        % normalize probabilities and find most probable
                    [P,init{8}]=maxP(allP);     %normalizes and outputs probability to screen
                    for zz=1:mC
                            result{3,zz}=P(zz);     %stores norm. prob. for each model
                    end
%----------assigning arrays for each optimality------------
 popt=init;presult=result;        dopt=init;gopt=init;dresult=result; gresult=result;%assign cell arrays to correct optimality for storage%%
%%       loop for pfun************************************
 fprintf('starting p-optimal experimental loop \n');
%start4=datestr(now)
 count=0;          %preset to zero
OPTIONS=optimset('Display','iter', 'LargeScale', 'off');
for i=1:ni
        popt{7}=i;
        iterp=i
        best=popt{8}(i);          %stores the number of most probable model for this iteration
        if flag2(2)==1  %flag2 tells whether or not to run box algorithm (1=yes)
                    [pointp,XY4,Z4,Z1]=optgraph(popt{4}(1,:),popt{4}(2,:),step1,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i), 4);  %This is used to generate XY and Z for box1.m inputs
                    [pout{i},popt{15}(i,:),popt{9}(i)]=box1({XY4{:}},Z4,Z1,{popt{:}},{presult{:,best}},{model{:,best}},2);
                    [LB1,UB1]=gridint(popt{15}(i,:),Low2,Up2,step1);
                    [popt{5,1}(i,:),fnew] = fmincon(@(xd)optfun(xd,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i),2,sc2),popt{15}(i,:),[],[],[],[],LB1,UB1);
    else                              %
                   %first, find best starting point for optimization using optgraph
                    [pointp2(i,:),XY5, Z5(:,:,i),Z6]=optgraph(popt{4}(1,:),popt{4}(2,:),step1, presult{1,best}(i,:),{model{:,best}}, {popt{:}},{presult{:,best}},presult{11,best}(i),2);
                    [popt{5}(i,:),fnew,Gp] = fmincon(@(xd)optfun(xd,presult{1,best}(i,:),{model{:,best}},{popt{:}},{presult{:,best}},presult{11,best}(i),2,sc2),pointp2(i,:),[],[],[],[],popt{4}(:,1),popt{4}(:,2),[],OPTIONS);
   end
        popt{17}(:,i)=scale(popt{5}(i,:),1);    %stores descaled values of experimental points
        %generate new data point
                    nisl1=nislcalc(popt{5}(i,:));
                    popt{3}(i+n)=nisl1;
    if model{7,best}==1
                    [newGS]=feval(model{1,best},popt{3}(i+n),presult{1,best}(i,:)); %gets model prediction of n_{isl}, different calculation if using hybrid models
        else
                    [newGS]=feval(model{1,best},popt{5}(i,:),presult{1,best}(i,:)); %gets model prediction of n_{isl}
        end
        [data5]=[popt{5}(i,:) newGS];
        popt{2}=[popt{2};data5];
    for j=1:mC
            %calculate metrics for each model
            [presult{1,j}(i+1,:),presult{2,j}(i+1),presult{4,j}(i+1,:),presult{5,j},presult{8,j}(i+1),presult{11,j}(i+1),presult{12,j}(i+1),popt{1,1}(i+1),presult{16,j}(i+1),presult{18,j}(i+1),presult{21,j}(i+1,:),presult{23,j}]=metrix3({model{:,j}},{popt{:,1}},{presult{:,j}},step2);
            presult{17,j}(:,i+1)=scale(presult{4,j}(i+1,:),1);  %descales T and C
            allP(j)=mprob(iprob,model{2,j}, popt{6}, presult{2,j}(i+1));     %calculate probabilities for each model
    end
     %calculate normalized probabilities for each model and find most
     %probable model
    [P,popt{8,1}(i+1)]=maxP(allP);
    for j=1:mC
            %assign probabilities to the respective model arrays
            presult{3,j}(i+1)=P(j);
     end
     exper=size(popt{2});      %calculates how many experiments have been run
     [pout1(i+1,:),count,popt{11}(i+1,:)]=SC2(popt{7},{presult{:,popt{8}(i+1)}},goal,count,exper(1),rep,popt{16});
     if pout1(i+1)==2
           if flag2(4)==1
               break
           else
               pout1(i+1)=0;           %if SC is not being run, change out to zero so box algorithm can be run
           end
     end
end
%--------------------------Saving Data--------------
        p=datestr(now,'mm_dd_yy');
        ch5=[p,'_',num2str(z),'_',num2str(num)];
        save(ch5);
clear point
count=0;        %preset to zero
%**************************D-optimal*****************
fprintf('starting d-optimal experimental loop \n');
        for i=1:ni
                dopt{7}=i;
                iterd=i
                best=dopt{8}(i);        %stores the number of most probable model for this iteration
                if  flag2(2)==1 %flag2 tells whether or not to run box algorithm (1=yes)
                        [pointd,XY4,Z4,Z1]=optgraph(dopt{4}(1,:),dopt{4}(2,:),step1,dresult{1,best}(i,:), {model{:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best}(i), 4); %This is used to generate XY and Z for box1.m inputs
                        [dout{i},dopt{15}(i,:),dopt{9}(i)]=box1({XY4{:}},Z4,Z1,{dopt{:}},{dresult{:,best}},{model{:,best}},1);
                        [LB1,UB1]=gridint(dopt{15}(i,:),Low2,Up2,step1);
                        [dopt{5,1}(i,:),fnew] = fmincon(@(xd)optfun(xd,dresult{1,best}(i,:),{model{:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best}(i),1,sc2),dopt{15}(i,:),[],[],[],[],LB1,UB1);
                else                          %run normal optimazation routine
                      %first , find best starting point for optimization using optgraph
```

```
                              [pointd2(i,:),XY5, Z5(:,:,i),Z6]=optgraph(dopt{4}(1,:),dopt{4}(2,:),step1,
                                  dresult{1,best}(i,:),{model{:,best}}, {dopt{:}},{dresult{:,best}},
                                dresult{11,best}(i),1);
                              [dopt{5}(i,:),fnew] = fmincon(@(xd)optfun(xd,dresult{1,best}(i,:),{model
                                {:,best}},{dopt{:}},{dresult{:,best}},dresult{11,best}(i),1,sc2),pointd2
                                (i,:) ,[],[],[],[],dopt{4}(:,1),dopt{4}(:,2),[],OPTIONS);
                    end
                    dopt{17}(:,i)=scale(dopt{5}(i,:),1);    %stores descaled values of experimental
                       points
          %generate new data point
                    nisl1=nislcalc(dopt{5}(i,:));
                    dopt{3}(i+n)=nisl1;
      if model{7,best}==1
                    [newGS]=feval(model{1,best},dopt{3}(i+n),dresult{1,best}(i,:)); %gets model
                       prediction of n_{isl}
          else
                    [newGS]=feval(model{1,best},dopt{5}(i,:),dresult{1,best}(i,:)); %gets model
                       prediction of n_{isl}
          end
          [data5]=[dopt{5}(i,:) newGS];
          dopt{2}=[dopt{2};data5];
          for j=1:mC
                     %calculate metrics for each model
                     [dresult{1,j}(i+1,:),dresult{2,j}(i+1),dresult{4,j}(i+1,:),dresult{5,j},dresult
                        {8,j}(i+1),dresult{11,j}(i+1),dresult{12,j}(i+1),dopt{1}(i+1),dresult{16,j}(i
                        +1),dresult{18,j}(i+1),dresult{21,j}(i+1,:),dresult{23,j}]=metrix3({model{:,j
                        }},{dopt{:,1}},{dresult{:,j}},step2);
                         dresult{17,j}(:,i+1)=scale(dresult{4,j}(i+1,:),1);  %descales T and C
                     allP(j)=mprob(iprob,model{2,j}, dopt{6}, dresult{2,j}(i+1));     %calculate
                        probabilities for each model
          end
          %calculate normalized probabilities for each model and find most
           %probable model
          [P,dopt{8,1}(i+1)]=maxP(allP);
          for j=1:mC
                     %assign probabilities to the respective model arrays
                     dresult{3,j}(i+1)=P(j);
          end
          exper=size(dopt{2});      %calculates how many experiments have been run
          [dout1(i+1,:),count,dopt{11}(i,:)]=SC2(dopt{7},{dresult{:,dopt{8}(i+1)}},goal,count,exper
            (1),rep,dopt{16});
          if dout1(i+1)==2
                    if flag2(4)==1
                              break
                    else
                              dout1(i+1)=0;           %if SC is not being run, change out to zero so box
                                  algorithm can be run
                    end
          end
end
%---------------------------Saving Data------------
          p=datestr(now,'mm_dd_yy');
          ch5=[p,'_',num2str(z),'_',num2str(num)];
          save(ch5);
clear point
```

```
count=0;        %preset to zero
%**********************RANDOM********************
fprintf('starting random experimental loop \n');
for i=1:ni
        gopt{7}=i;
        iterg=i
        best=gopt{8}(i);        %stores the number of most probable model for this iteration
        if flag2(2)==1  %flag2 tells whether or not to run box algorithm (1=yes)
                [pointg,XY4,Z4,Z1]=optgraph(gopt{4}(1,:),gopt{4}(2,:),step1,gresult{1,best}(i,:),{model{:,best}},{gopt{:}},{gresult{:,best}},gresult{11,best}(i), 4);  %This is used to generate XY and Z for box1.m inputs
                [gout{i},gopt{15}(i,:),gopt{9}(i)]=box1({XY4{:}},Z4,Z1,{gopt{:}},{gresult{:,best}},{model{:,best}},4);
                [o,q]=size(gout{i});
                if q>=8
                        pop=round(unifrnd(1,o));
                        gopt{5,1}(i,:)=gout{i}(pop,2:3);
                else                    %if points can't be found on the grid, then make random points
                        gopt{5,1}(i,1) = unifrnd(gopt{4}(1,1),gopt{4}(1,2));
                        gopt{5,1}(i,2)= unifrnd(gopt{4}(2,1),gopt{4}(2,2));
                end
        else
                %generate experimental points using random number generator, unifrnd picks random number between optrange
                gopt{5,1}(i,1) = unifrnd(gopt{4}(1,1),gopt{4}(1,2));
                gopt{5,1}(i,2)= unifrnd(gopt{4}(2,1),gopt{4}(2,2));
        end
        gopt{17}(:,i)=scale(gopt{5}(i,:),1);    %stores descaled values of experimental points
        %generate new data point
                nisl1=nislcalc(gopt{5}(i,:));
                gopt{3}(i+n)=nisl1;
    if model{7,best}==1
                [newGS]=feval(model{1,best},gopt{3}(i+n),gresult{1,best}(i,:)); %gets model prediction of n_{isl}
        else
                [newGS]=feval(model{1,best},gopt{5}(i,:),gresult{1,best}(i,:)); %gets model prediction of n_{isl}
        end
        [data5]=[gopt{5}(i,:) newGS];
        gopt{2}=[gopt{2};data5];
        for j=1:mC
                 %calculate metrics for each model
                 [gresult{1,j}(i+1,:),gresult{2,j}(i+1),gresult{4,j}(i+1,:),gresult{5,j},gresult{8,j}(i+1),gresult{11,j}(i+1),gresult{12,j}(i+1),gopt{1,1}(i+1),gresult{16,j}(i+1),gresult{18,j}(i+1),gresult{21,j}(i+1,:),gresult{23,j}]=metrix3({model{:,j}},{gopt{:,1}},{gresult{:,j}},step2);
                  gresult{17,j}(:,i+1)=scale(gresult{4,j}(i+1,:),1);  %descales T and C
                 allP(j)=mprob(iprob,model{2,j}, gopt{6}, gresult{2,j}(i+1));     %calculate probabilities for each model
         end
        %calculate normalized probabilities for each model and find most
         %probable model
         [P,gopt{8,1}(i+1)]=maxP(allP);
         for j=1:mC
```

```
                %assign probabilities to the respective model arrays
                gresult{3,j}(i+1)=P(j);
        end
        exper=size(gopt{2});      %calculates how many experiments have been run
        [gout1(i+1,:),count,gopt{11}(i+1,:)]=SC2(gopt{7},{gresult{:,gopt{8}(i+1)}},goal,count,
          exper(1),rep,gopt{16});
        if gout1(i+1,1)==2
                  if flag2(4)==1
                        break
                  else
                        gout1(i+1,1)=0;                %if SC is not being run, change out to
                          zero so box algorithm can be run
                  end
        end
end
clear point
finish=datestr(now)
%—————————————————————————————Saving Data——————————————
        p=datestr(now,'mm_dd_yy');
        ch5=[p,'_',num2str(z),'_',num2str(num)];
        save(ch5);
%% extra functions needed
%%%%%%%%%%%%%%%%%%%%%%%%%%%%box1%%%%%%%%%%%%%%%%%%%%%%
function [out,point,boxmax]=box1(mat,Z,Zall,opt,result,model8,flag)
%***************modified for nucleation study*******
%this function takes as input the array of x and y values 'X', and the
%matrix of some functions evaluations in the (x,y) space, Z.  'boxmax' is the
%maximum allowed value calculated from avareage output + CI
%Z1 is the model evaluations at (X,Y) where Z is the objective function evaluations at (X,Y)
%————————Constraints———————————————————————————
global ntarget conf2
%——————————————————————————————————————————————
a=opt{7};        %iteration
counter=0;       %need something to track how many points are below boxmax
theta=result{1}(a,:);   %parameters
sigsq=result{11}(a);    %exp error
hand=model8{1};         %handel for model function
d=opt{1}(a);            %max value of data for Jac.m
J=result{5};            %X matrix (derivative of model wrt parameters)
n=length(opt{2});       %calculates how many experiments have been performed
delta=result{12}(a);    %confidence interval on F
p=model8{2};            %# parameters in model
nisl=result{18}(a);
value=result{16}(a);    %value of obj. func. at est. opt. point of most probable model
%Calculation of variance on objective function
CI=result{21}(a,2);     %confidence interval of objective function calculated in metrix3 for
   optimal point
boxmax=value+CI;        %
X=mat{1};        Y=mat{2};
Z2=Zall{1};      %taking Zall apart to be used by box.m function (Z1=time, Z2=nisl, Z3=F, Z4=dnisl)
for i=1:length(X)
        for j=1:length(X)
                point=[X(i,j) Y(i,j)];
                %calculate prediction variance at point
                if model8{7}==0,
                    Jopt=Jac(theta,hand,point,d);   %calculates Jacobian of ALL the data to use in D-optimal.
                else
                    Jnisl1=nislcalc(point);
                    Jopt=Jac(theta,hand,Jnisl1,d);
                end
                PVopt1=PV(J,Jopt,sigsq);
                %calculate confidence interval at estimated optimal point.  Equation taken
                %from pg 395 Montgomery.  **Jopt should be a 3x1 matrix, so the transpose is
                %reversed**
                delta2=tinv(conf2,(n(1)-p))*sqrt(sigsq*(Jopt*inv(J'*J)*Jopt'));
                CI4=sqrt(2*abs((Z2(i,j)-ntarget)))*delta2;      %include both terms in objective function since both depend on F
                if (Z(i,j) < boxmax) || ((Z(i,j)-CI4) < boxmax)    %want value to be greater than boxmax since the point is to maximize N_isl
                    counter=counter+1;
                    out(counter,:)=[CI4 X(i,j) Y(i,j) Z(i,j) Z2(i,j) CI boxmax delta2];      %the column indice of Z is the x indice, row is for y indice
                end
        end
end
if counter==0           %if nothing was found that met contstraints, end function
   out=1;               %out returns a dummy variable so no error
   return
end
%out is of the form [output x y] for as many rows as values below 'boxmax'
% part 2 of function-picking best value for pfun
[n,m]=size(out);
if flag==4              %if flag==4, doing random and want to skip this set of calculations
        point=[out(1,2) out(1,3)];      %if flag==4, this point is meaningless
else
        for k=1:counter       %calc value of optimality (depends on flag) at the points below Pvalue
                out(k,9)=optfun([out(k,2) out(k,3)],theta,model8,opt,result,sigsq,flag,1);
        end
        %-----------FINDING MINIMUM-------------------
        [val,I]=min(out(:,9));          %finds minimum of each column in 'b' and its indice
        xfin=out(I,2);
        yfin=out(I,3);
        point=[xfin yfin];       %index of minimum point
end
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%metrix3%%%%%%
function [theta,Sr,point,J,PVopt1,sigmasq,CI,d,val,nisl,CI5,Jopt]=metrix3(model2,opt,result,step)
%function
% metrix3 is to calculate the model outputs for the roughness models, also calculates J for time models as well and propagates error for both models
%*************Modified for nucleation simulation**
%This function will reduce the clutter in the main function.  This function
%will calculate the D-optimal value
%%%%INPUT Definitions
%model2=array passed with needed information for optimality
%err=experimental error
%range=lower and upper bounds for calculating new thetas
%optrange=range for finding optimum point
```

```
%step=how many steps one wants optgraph to perform
%%%%OUTPUT Definitions
%theta=predicted thetahat for this iteration
%Sr=predicted Srhat for this iteration
%d=predicted max value of y from data
%J=predicted jacobian for this iteration
%data=data matrix plus new experiment from this iteration
%Dopt=D-optimal value for this iteration
%PVx1=prediction variance for experimental point
%pred=predicted value at estimated optimal point
%err1=error of predicted value to actual value at predicted optimal point
%PVopt1=PV at initial estimated optimal point
%PVopt2=PV at real optimal point
%pred2=output from system
%sigmasq=experimental variance calculated from experimental data
%optpoint=estimated optimal point for this iteration (after new experiment
%is generated (used in calculations for next round
%-------------pulling information from arrays
a=opt{7};                 %passes # of iteration to find correct value
data=opt{2};              %experimental data
hand=model2{1};           %model function handle
exppt=opt{5}(a,:);        %next experimental point
xLB=opt{4}(:,1);          %range for x1
xUB=opt{4}(:,2);          %range for x2
p=model2{2};              % # parameters in model
rep=opt{6};          %number repititions
ndata1=data;
%calculate new thetahat and Srhat
global ntarget conf2
%----------------------------------------
%       These calculations need to be performed BEFORE running pinky, so here they are outside the
    while loop
        [theta, Sr, G] = newfit(ndata1, model2,opt{13},opt{3});
        n=size(ndata1);
        sigmasq=Sr/(n(1)-p);              %calculating experimental variance
        d=max(round(ndata1(:,3)*100))/100;                %calculates maximum of output for
          jacobian calculation
        %*1000 is used so d is a number other than 0
        if model2{7}==0,
                J=Jac(theta,hand,ndata1,d);   %calculates Jacobian of ALL the data to use in D-
                  optimal.
        else J=Jac(theta,hand,opt{3}',d);       %if hybrid model is most probable, change inputs
          to Jac
        end
        hand
        %find optimal point predicted by the optimality
        %first, use optgraph to start optimization
        [pointp,XY4,Z4,Z1]=optgraph(opt{4}(1,:),opt{4}(2,:),step, theta, {model2{:}},{opt{:}},{
          result{:}},sigmasq, 4);
        OPTIONS = optimset( 'Display', 'iter', 'LargeScale', 'off');
        [point,val]=fmincon(@(point)objfun(point,theta,{model2{:}},d),pointp,[],[],[],[],xLB,xUB);
        [val,stuff]=objfun(point,theta,{model2{:}},d);  %need to recalc objfun.m now that opt.
          point is found for other parameters needed for error propagation
        nisl=stuff;
        if model2{7}==0,
                Jopt=Jac(theta,hand,point,d);   %calculates Jacobian of ALL the data to use in D-
                  optimal.
        else
                Jnisl1=nislcalc(point);
                Jopt=Jac(theta,hand,Jnisl1,d);
        end
%       %calculate confidence interval at estimated optimal point.  Equation taken
%       %from pg 395 Montgomery.  **Jopt should be a 3x1 matrix, so the transpose is
%       %reversed**
        CI=tinv(conf2,(n(1)-p))*sqrt(sigmasq*(Jopt*inv(J'*J)*Jopt'));
%       %Calculate prediction at estimated optimal point
        CI4=sqrt(2*abs((nisl-ntarget)))*CI;     %include both terms in objective function since
          both depend on F
        CI5=[CI CI4];           %array to store all confidence intervals
        boxmax=val+CI4; %
        PVopt1=PV(J,Jopt,sigmasq);
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%nislcalc%%%%%%%%%%%%
function [dnisl1,data2]=nislcalc(u)
% Evans, Thiel, and Bartlett (2006) equation (6.4)
%This is the equation used to generate the data from experimental runs
%suggested by D(x)
%INPUTS
%u(1)=T   (scaled)
%u(2)=C   (scaled)
%z=# for saving output graph
%rep=# of repititions to be done
%err=variance of gaussian data
%------using global variables
global global_conv2 global_thick Ei Ed sigma thetaf
%---------------Un-Scaling variables-------------
z=scale(u,1);
T = z(1); %temperature [K]
F=3341.07*z(2);          %[ML/micromol] conversion factor of molar flow to ML/min
x=[T F];
        %-----------PARAMETERS%----------------
        q=[Ei Ed sigma];        %stores Ei,E_d, and sigma in array to pass to nuc
        n0 = [0; 0; 0];  % Start with a bare Si surface
        tf = thetaf/F;    % Final time to reach thetaf
        dt = tf/100;    %size of time step
        trange = (0:dt:tf);  % Range of times [s]
                [t,n] = ode23t(@nuc,trange,n0,[],x,q);
                data2(1,:)=[n(end,3) tf max(n(:,2)) F];
                twodnisl=data2(1,1);
                if twodnisl <=0
                        error('2D nisl is less than zero');
                end
                dnisl1=sqrt(twodnisl);  %take square root of 2D nisl for input to hybrid models
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%nuc%%%%%%%%%%%%
function dndt = nuc(t,n,x,q)
% Nucleation model from Evans, Thiel, and Bartelt
%global variables-------------
global kb global_v ci global_i
% Process inputs (factors)
```

```
T = x(1);
F=x(2);
%q are the fitted parameters in this model
    Ei = q(1); % Energy of critical cluster [eV]
    Ed = q(2); % diffusion activation energy [eV]
    sigma = q(3);   % Capture number
beta =1/kb/T; %[1/eV]
h = global_v*exp(-beta*Ed);  %[1/s]
%calculation of N_isl
% States
theta = n(1);
N1 = n(2);
Nisl = n(3);
Ni = ci*exp(-beta*Ei)*N1^global_i; % Density of critical clusters
Knuc = sigma*h*N1*Ni;
Kagg = sigma*h*N1*Nisl;
% Differential equation model
dndt = zeros(3,1);
dndt(1) = F;   % Coverage theta
dndt(2) = F*(1-theta) - (global_i+1)*Knuc - Kagg;
dndt(3) = sigma*h*N1*Ni;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%objfun%%%%%%%%%%%
function [z1,r]=objfun(x,theta,model,d)
%this function calculates the value of the objective function for the
%nucleation study
%--------Constraints-----------------------
global ntarget
%if you change these constraints, must also change constraints in box1.m
%------Calculation-------------------------
hand=model{1};
if model{7}==1
        nisl1=nislcalc(x);
        x=nisl1;
end
[GS]=feval(hand,x,theta);       %gets model prediction of roughness
z1=(((GS-ntarget))^2);
r=[GS];
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%Scale%%%%%%%%%%%%%%
function z=scale(x,flag)
%should output z as a column vector [T;C]
%flag =1 means descale, otherwise, scale variables
Ta=960.5;                Tb=87.5;
Ca=150; Cb=50;
if flag==1
        %this function descales the x values so they can be expressed in K and micromoles/min
          respectively
        z(1,1)=x(1)*Tb+Ta;               %Temp
        z(2,1)=x(2)*Cb+Ca;                         %concentration
else    %Scale variables
        %this function scales the x values so they can be expressed in K and micromoles/min
          respectively
        z(1,1)=(x(1)-Ta)/Tb;             %Temp
        z(2,1)=(x(2)-Ca)/Cb;                       %concentration
```

```
end
end
%%%%%%%%%%%%%%%%%%%%%%rms1%%%%%%%%%%%%%%%%%%%
function [GS]=rms1(x,q)
%T=temperature [K]
%These are the dependant variables
%(need to be descaled for mechanistic model)
%u=scale(x,1);
T = x(1); %temperature [K]
C = x(2); %Concentration [micromoles/L]
a=q(1);
b=q(2);
GS=a+b*T;
end
%%%%%%%%%%%%%%%%%%%%%%rms2%%%%%%%%%%%%%%%%%
function [GS]=rms2(x,q)
%T=temperature [K]
%These are the dependant variables
%(need to be descaled for mechanistic model)
%u=scale(x,1);
T = x(1); %temperature [K]
C = x(2); %Concentration [micromoles/L]
a=q(1);
b=q(2);
c=q(3);
GS=a+b*T+c*C;
end
%%%%%%%%%%%%%%%%%%%%%%rms3%%%%%%%%%%%%%%%
function [GS]=rms3(x,q)
%T=temperature [K]
%These are the dependant variables
%(need to be descaled for mechanistic model)
a=q(1);
%u=scale(x,1);
%T = x(1); %temperature [K]
%C = x(2); %Concentration [micromoles/L]
b=q(2);
GS=a+b*x;
end
%%%%%%%%%%%%%%%%%%%%%%rms4%%%%%%%%%%%%%
function [GS]=rms4(x,q)
%T=temperature [K]
%These are the dependant variables
%(need to be descaled for mechanistic model)
%u=scale(x,1);
T = x(1); %temperature [K]
C = x(2); %Concentration [micromoles/L]
a=q(1);
b=q(2);
c=q(3);
GS=a+b*T+c*T*C;
end
%%%%%%%%%%%%%%%%%%%%%%rms5%%%%%%%%%%%%
function [GS]=rms5(x,q)
%T=temperature [K]
```

```
%These are the dependant variables
%(need to be descaled for mechanistic model)
%u=scale(x,1);
T = x(1); %temperature [K]
C = x(2); %Concentration [micromoles/L]
a=q(1);
b=q(2);
GS=a+b*T*C;
end
%%%%%%%%%%%%%%%%%%%%%%rms6%%%%%%%%%%%
function [GS]=rms6(x,q)
%T=temperature [K]
%These are the dependant variables
%(need to be descaled for mechanistic model)
a=q(1);
GS=a;
end
%%%%%%%%%%%%%%%%%%%%rms7%%%%%%%%%%%%%
function [GS]=rms7(x,q)
%x is the input of nucleation density
%q is the fitted parameter
GS=q*x;
end
```

Listing D.3: exp7.m in `Matlab`

REFERENCES

[1] Ahmed, S. and Sahinidis, N., "Robust process planning under uncertainty," *Industrial & Engineering Chemistry Research*, vol. 37, no. 5, pp. 1883–1892, 1998.

[2] Alford, J. S., "Bioprocess control: Advances and challenges," *Computers & Chemical Engineering*, vol. 30, no. 10-12, pp. 1464–1475, 2006.

[3] Allen, J. K., Seepersad, C., Choi, H., and Mistree, F., "Robust design for multiscale and multidisciplinary applications," *Transactions of the ASME*, vol. 128, pp. 832–843, 2006.

[4] Arkun, Y., Manousiouthakis, B., and Palazoglu, A., "Robustness analysis of process control systems. a case study of decoupling control in distillation," *Industrial & Engineering Chemistry Process Design and Development*, vol. 23, pp. 93–101, 1984.

[5] Aspnes, D., Theeten, J., and Hottier, F., "Investigation of effective-medium models for microscopic surface roughness by spectroscopic ellipsometry," *Physical Review B*, vol. 20, pp. 3292–3302, 1979.

[6] Asprey, S. and Macchietto, S., "Statistical tools for optimal dynamic model building," *Computers & Chemical Engineering*, vol. 24, no. 2-7, pp. 1261–1267, 2000.

[7] Atkinson, A. C. and Fedorov, V. V., "The design of experiments for discriminating between two rival models," *Biometrika*, vol. 62, pp. 57–70, 1975.

[8] Atkinson, A. C. and Fedorov, V. V., "Optimal design: Experiments for discriminating between several models," *Biometrika*, vol. 62, pp. 289–303, 1975.

[9] Atkinson, A., "Planning experiments to detect inadequate regression models," *Biometrika*, vol. 59, pp. 275–293, 1972.

[10] Aughenbaugh, J. M. and Paredis, C. J., "The value of using imprecise probabilities in engineering design," *Journal of Mechanical Design*, vol. 128, pp. 969–979, 2006.

[11] Axtell, N. R., Sternberg, S. P., and Claussen, K., "Lead and nickel removal using microspora and lemna minor," *Bioresource Technology*, vol. 89, pp. 41–48, 2003.

[12] Bao, T., Morrison, P. W., and Woyczynski, W. A., "Parametric optimization of microhardness of diamond-like carbon films prepared by plasma enhanced chemical vapor deposition," *Thin Solid Films*, vol. 485, no. 1-2, pp. 27–41, 2005.

[13] Barreca, D., Armelao, L., Caccavale, F., Noto, V. D., Gregori, A., Rizzi, G. A., and Tondello, E., "Highly oriented V_2O_5 nanocrystalline thin films by plasma-enhanced chemical vapor deposition," *Chem. Mater.*, vol. 12, pp. 98–103, 2000.

[14] Bays, H., Ose, L., Fraser, N., Tribble, D., Quinto, K., Reyes, R., Johnson-Levonas, A., Sapre, A., Donahue, S., and Ezetimibe Study Grp, "A multicenter, randomized, double-blind, placebo-controlled, factorial design study to evaluate the lipid-altering efficacy and safety profile of the ezetimibe/simvastatin tablet compared with ezetimibe and simvastatin monotherapy in patients with primary hypercholesterolemia," *Clinical Therapeutics*, vol. 26, no. 11, pp. 1758–1773, 2004.

[15] Bernardo, J. M., "Expected information as expected utility," *The Annals of Statistics*, vol. 7, pp. 686–690, 1979.

[16] Biscarini, F., Zamboni, R., Samori, P., Ostoja, P., and Taliani, C., "Growth of conjugated oligomer thin films studied by atomic-force microscopy," *Physical Review B*, vol. 52, pp. 14868–14877, 1995.

[17] Box, G. and Kramer, T., "Statistical process monitoring and feedback adjustment - a discussion," *Technometrics*, vol. 34, no. 3, pp. 251–267, 1992.

[18] Box, G. E. P. and Hill, W. J., "Discrimination among mechanistic models," *Technometrics*, vol. 9, no. 1, p. 57, 1967.

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

[20] Braake, H. A. B., van Can, H. J. L., and Verbruggen, H. B., "Semi-mechanistic modeling of chemical processes with neural networks," *Engineering Applications of Artificial Intelligence*, vol. 11, no. 4, pp. 507–515, 1998.

[21] Breiland, W., "Reflectance-correcting pyrometry in thin film deposition applications," tech. rep., Sandia National Laboratories, 2003.

[22] Breiland, W. G. and Evans, G. H., "Design and verification of nearly ideal flow and heat transfer in a rotating disk chemical vapor deposition reactor," *Journal of the Electrochemical Society*, vol. 138, pp. 1806–1816, 1991.

[23] Brewer, R. T., Boyd, D. A., El-Naggar, M. Y., Boland, S. W., Park, Y.-B., Haile, S. M., Goodwin, D. G., and Atwater, H. A., "Growth of biaxially textured $Ba_xPb_{1-x}TiO_3$ ferroelectric thin films on amorphous Si_3N_4," *Journal of Applied Physics*, vol. 97, pp. 1–8, 2004.

[24] Buzzi-Ferraris, G. and Forzatti, P., "A new sequential experimental design procedure for discriminating among rival models," *Chemical Engineering Science*, vol. 38, no. 2, pp. 225–232, 1983.

[25] Carmona, M., da Silva, M., and Leite, S., "Biosorption of chromium using factorial experimental design," *Process Biochemistry*, vol. 40, no. 2, pp. 779–788, 2005.

[26] Carniglia, C. and Jensen, D., "Single-layer model for surface roughness," *Applied Optics*, vol. 41, pp. 3167–3171, 2002.

[27] Chaloner, K., "Optimal Bayesian experimental-design for linear-models," *Annals Of Statistics*, vol. 12, no. 1, pp. 283–300, 1984.

[28] Chaloner, K. and Verdinelli, I., "Bayesian experimental design: A review," *Statistical Science*, vol. 10, no. 3, pp. 273–304, 1995.

[29] Chang, C. and Kryder, M., "Effect of substrate roughness on microstructure, uniaxial anisotropy, and coercivity of Co/Pt multilayer thin-films," *Journal Of Applied Physics*, vol. 75, no. 10, Part 2B, pp. 6864–6866, 1994.

[30] Chapra, S. C. and Canale, R. P., *Numerical Methods for Engineers.* McGraw-Hill, 4th ed., 2002.

[31] Chen, B. H. and Asprey, S., "On the design of optimally informative dynamic experiments for model discrimination in multiresponse nonlinear situations," *Industrial & Engineering Chemistry Research*, vol. 42, pp. 1379–1390, 2003.

[32] Chen, J. H., Wong, D. S. H., Jang, S. S., and Yang, S. L., "Product and process development using artificial neural-network model and information analysis," *AIChE Journal*, vol. 44, no. 4, pp. 876–887, 1998.

[33] Chou, K. S. and Tsai, G. J., "Dynamic evaporation behavior of diketonate compounds of yttrium, copper and barium," *Thermochimica Acta*, vol. 240, pp. 129–139, 1994.

[34] Choy, K. L., "Chemical vapour deposition of coatings," *Progress in Materials Science*, vol. 48, no. 2, pp. 57–170, 2003.

[35] Coleman, M. C. and Block, D. E., "Nonlinear experimental design using Bayesian regularized neural networks," *AIChE Journal*, vol. 53, pp. 1496–1509, 2007.

[36] Coumes, C. C. D. and Courtois, S., "Design of experiments to investigate main effects and all two-factor interactions when one of the factors has more than two levels-application to nuclear waste cementation," *Chemometrics And Intelligent Laboratory Systems*, vol. 80, no. 2, pp. 167–175, 2006.

[37] Currin, C., Mitchell, T., Morris, M., and Ylvisaker, D., "Bayesian prediction of deterministic fucntions, with applications to the design and analysis of computer experiments," *Journal of the American Statistical Association*, vol. 86, pp. 953–963, 1991.

[38] DeSisto, W. J. and Rappoli, B. J., "Ultraviolet absorption sensors for precursor delivery rate control for metalorganic chemical vapor deposition of multiple component oxide thin films," *Journal Of Crystal Growth*, vol. 191, no. 1-2, pp. 290–293, 1998.

[39] Dette, H., Haines, L. M., and Imhof, L., "Optimal designs for rational models and weighted polynomial regression," *The Annals of Statistics*, vol. 27, pp. 1272–1293, 1999.

[40] Diwekar, U. M. and Rubin, E. S., "Parameter design methodology for chemical processes using a simulator," *Industrial & Engineering Chemistry Research*, vol. 33, no. 2, pp. 292–298, 1994.

[41] Dumouchel, W. and Jones, B., "A simple bayesian modification of d-optimal designs to reduce dependence on an assumed model," *Technometrics*, vol. 36, no. 1, pp. 37–47, 1994.

[42] Edgar, T. F., Butler, S. W., Campbell, W. J., Pfeiffer, C., Bode, C., Hwang, S. B., Balakrishnan, K. S., and Hahn, J., "Automatic control in microelectronics manufacturing: Practices, challenges, and possibilities," *Automatica*, vol. 36, no. 11, pp. 1567–1603, 2000.

[43] Evans, J., P.A., T., and Bartelt, M., "Morphological evolution during epitaxial thin film growth: Formation of 2d islands and 3d mounds," *Surface Science Reports*, vol. 61, pp. 1–128, 2006.

[44] Franceschini, G. and Macchietto, S., "Validation of a model for biodiesel production through model-based experiment design," *Industrial & Engineering Chemistry Research*, vol. 46, no. 1, pp. 220–232, 2007.

[45] Franta, D. and Ohlidal, I., "Comparison of effective medium approximation and Rayleigh-Rice theory concerning ellipsometric characterization of rough surfaces," *Optics Communications*, vol. 248, pp. 459–467, 2005.

[46] Friedrich, L. J., Dew, S. K., Brett, M., and Smy, T., "Thin-film microstructure modeling through line-segment simulation," *Thin Solid Films*, vol. 266, no. 1, pp. 83–88, 1995.

[47] Froment, G. F., "Model discrimination and parameter estimation in heterogeneous catalysis," *AIChE Journal*, vol. 21, no. 6, pp. 1041–1057, 1975.

[48] Fujiwara, M., Nagy, Z. K., Chew, J. W., and Braatz, R. D., "First-principles and direct design approaches for the control of pharmaceutical crystallization," *Journal Of Process Control*, vol. 15, no. 5, pp. 493–504, 2005.

[49] Fulem, M., Ruzicka, K., Ruzicka, V., Simecek, T., Hulicius, E., and Pangrc, J., "Vapour pressure and heat capacities of metal organic precursors, Y(thd)3 and Zr(thd)4," *Journal of Crystal Growth*, vol. 264, no. 1-3, pp. 192–200, 2004.

[50] Gorla, C., Emanetoglu, N., Liang, S., Mayo, W., Lu, Y., Wraback, M., and Shen, H., "Structural, optical, and surface acoustic wave properties of epitaxial ZnO films grown on (0112) sapphire by metalorganic chemical vapor deposition," *Journal of Applied Physics*, vol. 85, pp. 2595–2602, 1999.

[51] Graves, D. and Kushner, M., "Influence of modeling and simulation on the maturation of plasma technology: Feature evolution and reactor design," *Journal Of Vacuum Science & Technology A*, vol. 21, no. 5, Suppl. S, pp. S152–S156, 2003.

[52] Green, M., Gusev, E., Degraeve, R., and Garfunkel, E., "Ultrathin (¡ 4 nm) SiO2 and Si-O-N gate dielectric layers for silicon microelectronics: Understanding the processing, structure, and physical and electrical limits," *Journal Of Applied Physics*, vol. 90, no. 5, pp. 2057–2121, 2001.

[53] Gunawan, R., Ma, D. L., Fujiwara, M., and Braatz, R. D., "Identification of kinetic parameters in multidimensional crystallization processes," *International Journal Of Modern Physics B*, vol. 16, no. 1-2, pp. 367–374, 2002.

[54] Heinz, W. F. and Hoh, J. H., "Getting physical with your chemistry: Mechanically investigating local structure and properties of surfaces with the atomic force microscope," *Journal of Chemical Education*, vol. 82, pp. 695–703, 2005.

[55] Helton, J., "Treatment of uncertainty in performance assessments for complex-systems," *Risk Analysis*, vol. 14, pp. 483–511, 1994.

[56] Henson, M. A., "Nonlinear model predictive control: Current status and future directions," *Computers & Chemical Engineering*, vol. 23, pp. 187–202, 1998.

[57] Herney-Ramirez, J., Lampinen, M., Vicente, M. A., Costa, C. A., and Madeira, L. M., "Experimental design to optimize the oxidation of orange II dye solution using a clay-based fenton-like catalyst," *Industrial & Engineering Chemistry Research*, vol. 47, pp. 284–294, 2008.

[58] Houtman, C., Graves, D. B., and Jensen, K. F., "CVD in stagnation point flow," *Journal of the Electrochemical Society*, vol. 133, pp. 961–970, 1986.

[59] Imaishi, N., Sato, T., Kimura, M., and Akiyama, Y., "Micro/macro modeling of CVD synthesis," *J Cryst Growth*, vol. 180, no. 3-4, pp. 680–690, 1997.

[60] ITRS, "International technology roadmap for semiconductors," tech. rep., ITRS, 2007.

[61] Jongste, J., Oosterlaken, T., Bart, G., Janssen, G., and Radelaar, S., "Deformation of Si(100) wafers during rapid thermal annealing," *J. Appl. Phys.*, vol. 75, pp. 2830–2836, 1994.

[62] Joseph, V. R., "A Bayesian approach to the deisgn and analysis of fractionated experiments," *Technometrics*, vol. 48, pp. 219–229, 2006.

[63] Kennedy, M. C. and O'Hagan, A., "Bayesian calibration of computer models," *J. R. Statist. Society B*, vol. 63, pp. 425–464, 2001.

[64] Kuhfeld, W. F. and Tobias, R. D., "Large factorial designs for product engineering and marketing research applications," *Technometrics*, vol. 47, no. 4, pp. 532–532, 2005.

[65] Kurobe, K.-i., Ishikawa, Y., Yamamoto, Y., Fuyuki, T., and Matsunami, H., "Effects of grain boundaries in polycrystalline silicon thin-film solar cells based on the two-dimensional model," *Solar Energy Materials & Solar Cells*, vol. 65, pp. 201–209, 2001.

[66] Lee, K.-M., Rhee, C.-H., Kang, C. K., and Kim, J., "Sequential and simulataneous statistical optimization by dynamic design of experiment for peptide overexpression in recombinant escherichia coli," *Applied Biochemistry and Biotechnology*, vol. 135, pp. 59–80, 2006.

[67] Li, J. F., Liao, H. L., Ding, C., and Coddet, C., "Optimizing the plasma spray process parameters of yttria stabilized zirconia coatings using a uniform design of experiments," *Journal Of Materials Processing Technology*, vol. 160, no. 1, pp. 34–42, 2005.

[68] Li, J. and Steckl, A., "Nucleation and void formation mechanisms in SiC thin film growth on Si by carbonization," *J. Electrochem. Soc.*, vol. 142, pp. 634–641, 1995.

[69] Ling, J. M., Aughenbaugh, J. M., and Paredis, C. J., "Managing the collection of information under uncertainty using information economics," *Transactions of the ASME*, vol. 128, pp. 980–990, 2006.

[70] Liski, E. P., Luoma, A., and Zaigraev, A., "Distance optimality design criterion in linear models," *Metrika*, vol. 49, pp. 193–211, 1999.

[71] Liu, Q., Spanos, L., Zhao, C., and Irene, E., "Morphology study of the thermal-oxidation of rough silicon surfaces," *Journal of Vacuum Science & Technology A-Vacuum Surfaces and Films*, vol. 13, pp. 1977–1983, 1995.

[72] Liu, Y., Zha, S., and Liu, M., "Novel nanostructured electrodes for solid oxide fuel cells fabricated by combustion chemical vapor deposition (CVD)," *Advanced Materials*, vol. 16, no. 3, pp. 256–260, 2004.

[73] Lizama, C., Freer, J., Baeza, J., and Mansilla, H., "Optimized photodegradation of Reactive Blue 19 on TiO2 and ZnO suspensions," *Catalysis Today*, vol. 76, no. 2-4, pp. 235–246, 2002.

[74] Lopaeva, O. V., Hitchman, M. L., Shamlian, S. H., and Watson, D. R., "A study by GC-MS of the decomposition of precursors for the MOCVD of high temperature superconductors," *Journal de Physique IV*, vol. 12, pp. 4–85 to 4–92, 2002.

[75] Ma, D., Tafti, D., and Braatz, R., "Optimal control and simulation of multidimensional crystallization processes," *COMPUTERS & CHEMICAL ENGINEERING*, vol. 26, no. 7-8, pp. 1103–1116, 2002.

[76] Maozhi, L. and Evans, J., "Modeling of island formation during submonolayer deposition: A stochastic geometry-based simulation approach," *Multiscale Model Simul.*, vol. 3, pp. 629–657, 2005.

[77] Masi, M., Bertani, V., Cavallotti, C., and Carra, S., "Towards a multiscale approach to the growth of silicon films by chemical vapor deposition," *Materials Chemistry And Physics*, vol. 66, no. 2-3, pp. 229–235, 2000.

[78] Mason, M. S., Holt, J. K., and Atwater, H. A., "Quantitative modelling of nucleation kinetics in experiments for poly-Si growth on SiO_2 by hot wire chemical vapor deposition," *Thin Solid Films*, vol. 458, pp. 67–70, 2004.

[79] Matthews, H. B. and Rawlings, J. B., "Batch crystallization of a photochemical: Modeling, control, and filtration," *AIChE Journal*, vol. 44, pp. 1119–1127, 1998.

[80] Medeiros-Ribeiro, G., Bratkovski, A. M., Kamins, T. I., Ohlberg, D. A., and Williams, R. S., "Shape transition of germanium nanocrystals on a silicon (001) surface from pyramids to domes," *Science*, vol. 279, pp. 353–355, 1998.

[81] Medina, E. A., Venugopal, S., Frazier, W. G., Medeiros, S., Mullins, W. M., Chaudhary, A., Irwin, R. D., Srinivasan, R., and Malas, J. C., "Optimization of microstructure development: Application to hot metal extrusion," *Journal Of Materials Engineering And Performance*, vol. 5, no. 6, pp. 743–752, 1996.

[82] Meng, G., Song, H., Dong, Q., and Peng, D., "Application of novel aerosol-assisted chemical vapor deposition techniques for SOFC thin films," *Solid State Ionics*, vol. 175, pp. 29–34, 2004.

[83] Mohan, S. V., Rao, N. C., Prasad, K. K., Krishna, P. M., Rao, R. S., and Sarma, P., "Anaerobic treatment of complex chemical wastewater in a sequencing batch biofilm reactor: Process optimization and evaluation of factor interactions using the taguchi dynamic DOE methodology," *Biotechnology and Bioengineering*, vol. 90, pp. 732–745, 2005.

[84] Montgomery, D. C., *Design and Analysis of Experiments*, vol. 6th. John Wiley & Sons, Inc, 2005.

[85] Movchan, B. and Ta, M., "Multirow dislocation boundaries of grains in polycrystalline nickel," *Phys Metals Metallog*, vol. 27, no. 6, p. 180, 1969.

[86] Murarka, S., Gutmann, R., Kaloyeros, A., and Lanford, W., "Advanced multilayer metallization schemes with copper as interconnection metal," *Thin Solid Films*, vol. 236, no. 1-2, pp. 257–266, 1993.

[87] Murphy, E. F., Gilmour, S. G., and Crabbe, M. J. C., "Efficient and accurate experimental design for enzyme kinetics: Bayesian studies reveal a systematic approach," *Journal of Biochemical and Biophysical Methods*, vol. 55, pp. 155–178, 2003.

[88] Myers, Raymond H. Montgomery, D. C., *Response surface methodology : process and product optimization using designed experiments.* Wiley, 1995.

[89] Ni, D. and Christofides, P., "Dynamics and control of thin film surface microstructure in a complex deposition process," *Chem Eng Sci*, vol. 60, no. 6, pp. 1603–1617, 2005.

[90] Nourbakhsh, A., Ganjipour, B., Zahedifar, M., and Arzi, E., "Morphology optimization of CCVD-synthesized multiwall carbon nanotubes, using statistical design of experiments," *Nanotechnology*, vol. 18, pp. 1–7, 2007.

[91] Nyutu, E. K. and Suib, S. L., "Experimental design in the deposition of BN interface coatings on SiC fibers by chemical vapor deposition," *Surface & Coatings Technology*, vol. 201, pp. 2741–2748, 2006.

[92] Ott, R. L. and Longnecker, M., *An Introduction to Statistical Methods and Data Analysis.* Duxbury; Thomson Learning, 2001.

[93] Otway, D., Obi, B., and Rees Jr., W., "Precursors for chemical vapor deposition of yttrium barium copper oxide," *Journal of Alloys and Compounds*, vol. 251, pp. 254–263, 1997.

[94] Palik, E. D., ed., *Handbook of optical constants of solids.* Orlando: Academic Press, 1985.

[95] Park, C., Hwang, J. Y., Huang, M., and Anderson, T. J., "Investigation of an upflow cold-wall CVD reactor by gas phase Raman spectroscopy," *Thin Solid Films*, vol. 409, pp. 88–97, 2002.

[96] Pasko, S., Hubert-Pfalzgraf, L. G., and Abrutis, A., "Synthesis and characterization of hafnium *tert*-butylacetoacetate as new MOCVD precursor for HfO_2 films," *Materials Letters*, vol. 59, pp. 1836–1840, 2005.

[97] Pearson, P., "Nonlinear empirical modeling techniques," *Computers & Chemical Engineering*, vol. 30, pp. 1514–1528, 2006.

[98] Petrik, P., Biro, L., Fried, M., Lohner, T., Berger, R., Schneider, C., Gyulai, J., and Ryssel, H., "Comparative study of surface roughness measured on polysilicon using spectroscopic ellipsometry and atomic force microscopy," *Thin Solid Films*, vol. 315, pp. 186–191, 1998.

[99] Phadke, M., *Quality Engineering Using Robust Design.* P T R Prentice-Hall, Inc, 1989.

[100] Pistikopoulos, E. and Ierapetritou, M., "Novel-Approach For Optimal Process Design Under Uncertainty," *Computers & Chemical Engineering*, vol. 19, no. 10, pp. 1089–1110, 1995.

[101] Pomfret, M. B., Stoltz, C., Varughese, B., and Walker, R. A., "Structural and compositional characterization of yttria-stabilized zirconia: Evidence of surface-stabilized, low-valence metal species," *Analytical Chemistry*, vol. 77, pp. 1791–1795, 2005.

[102] Psichogios, D. C. and Ungar, L. H., "A hybrid neural network-first principles approach to process modeling," *AIChE Journal*, vol. 38, pp. 1499–1511, 1992.

[103] Pulver, M., Nemetz, W., and Wahl, G., "CVD Of ZrO2, Al2O3 And Y2O3 from metalorganic compounds in different reactors," *Surf Coat Tech*, vol. 125, no. 1-3, pp. 400–406, 2000.

[104] Pulver, M., Wahl, G., Scheytt, H., and Sommer, M., "Deposition of ZrO2 and Y2O3-stabilized ZrO2 from beta-diketonates," *Journal De Physique IV*, vol. 3, no. C3, pp. 305–312, 1993.

[105] Qian, Z. G., Seepersad, C. C., Joseph, V. R., Allen, J. K., and Wu, C. F. J., "Building surrogate models based on detailed and approximate simulations," *Journal of Mechanical Design*, vol. 128, no. 4, pp. 668–677, 2006.

[106] Rappoli, B. J. and DeSisto, W. J., "Gas phase ultraviolet spectroscopy of high-temperature superconductor precursors for chemical vapor deposition processing," *Applied Physics Letters*, vol. 68, no. 19, pp. 2726–2728, 1996.

[107] Ravikumar, K., Pakshirajan, K., Swaminathan, T., and Balu, K., "Optimization of batch process parameters using response surface methodology for dye removal by a novel adsorbent," *Chemical Engineering Journal*, vol. 105, pp. 131–138, 2005.

[108] Rawlings, J. B., Miller, S. M., and Witkowski, W. R., "Model identification and control of solution crystallization processes: a review," *Industrial & Engineering Chemistry Research*, vol. 32, no. 7, pp. 1275–1296, 1993.

[109] Rippin, D. W. T., "Statistical-methods for experimental planning in chemical-engineering," *Computers & Chemical Engineering*, vol. 12, no. 2-3, pp. 109–116, 1988.

[110] Robbins, J. J., Harvey, J., Leaf, J., Fry, C., and Wolden, C. A., "Transport phenomena in high performance nanocrystalline ZnO : Ga films deposited by plasma-enhanced chemical vapor deposition," *Thin Solid Films*, vol. 473, no. 1, pp. 35–40, 2005.

[111] Rodgers, S. and Jensen, K., "Multiscale modeling of chemical vapor deposition," *J Appl Phys*, vol. 83, no. 1, pp. 524–530, 1998.

[112] Rollett, A., Srolovitz, D., and Anderson, M., "Simulatino and theory of abnormal grain growth –anisotropic grain boundary energies and mobilities," *Acta Metall.*, vol. 37, pp. 1227–1240, 1989.

[113] Rubio, J. E., Jaraiz, M., Martin-Bragado, I., Hernandez-Mangas, J. M., Barbolla, J., and Gilmer, G. H., "Atomistic Monte Carlo simulations of three-dimensional polycrystalline thin films," *Journal Of Applied Physics*, vol. 94, no. 1, pp. 163–168, 2003.

[114] Savaloni, H. and Shahraki, M., "A computer model for the growth of thin films in a structure zone model," *Nanotechnology*, vol. 15, no. 3, pp. 311–319, 2004.

[115] Schugerl, K., "Progress in monitoring, modeling and control of bioprocesses during the last 20 years," *Journal Of Biotechnology*, vol. 85, no. 2, pp. 149–173, 2001.

[116] Schuler, T., Krajewski, T., Grobelsek, I., and Aegerter, M., "A microstructural zone model for the morphology of sol-gel coatings," *J Sol-Gel Sci Techn*, vol. 31, no. 1-3, pp. 235–239, 2004.

[117] Seshan, K., ed., *Handbook of thin-film deposition processes and techniques : principles, methods, equipment and applications*, vol. 2nd. Norwich, N.Y.: Noyes Publications/William Andrew Pub., 2002.

[118] Smith, D. L., *Thin-Film Deposition: Principles & Practice.* McGraw-Hill, Inc, 1995.

[119] Soyez, G., Eastman, J., Thompson, L., Bai, G., Baldo, P., McCormick, A., DiMelfi, R., Elmustafa, A., Tambwe, M., and Stone, D., "Grain-size-dependent thermal conductivity of nanocrystalline yttria-stabilized zirconia films grown by metal-organic chemical vapor deposition," *Appl Phys Lett*, vol. 77, no. 8, pp. 1155–1157–, 2000.

[120] Stein, A. and Ettema, C., "An overview of spatial sampling procedures and experimental design of spatial studies for ecosystem comparisons," *Agriculture Ecosystems & Environment*, vol. 94, no. 1, pp. 31–47, 2003.

[121] Stewart, W. E., Shon, Y., and Box, G. E. P., "Discrimination and goodness of fit of multiresponse mechanistic models," *AIChE Journal*, vol. 44, no. 6, pp. 1404–1412, 1998.

[122] Stewart, W. E., Caracotsios, M., and Sorenson, J. P., "Parameter estimation from multiresponse data," *AIChE Journal*, vol. 38, pp. 641–650, 1992.

[123] Stowell, M. and Hutchinson, T. E., "Nucleation kinetics in thin film growth ii analytical evaluation of nucleation and growth behaviour," *Thin Solid Films*, vol. 8, pp. 41–53, 1971.

[124] Studden, W. J., "Ds-optimal designs for polynomial regression using continued fractions," *Annals of Statistics*, vol. 8, no. 5, pp. 1132–1141, 1980.

[125] Swaney, R. and Grossmann, I., "An Index For Operational Flexibility In Chemical Process Design .1. Formulation And Theory," *AIChE Journal*, vol. 31, no. 4, pp. 621–630, 1985.

[126] Takors, R., Wiechert, W., and Weuster-Botz, D., "Experimental design for the identification of macrokinetic models and model discrimination," *Biotechnology and Bioengineering*, vol. 56, pp. 564–576, 1997.

[127] Thompson, C., "Structure evolution during processing of polycrystalline films," *Annu Rev Mater Sci*, vol. 30, pp. 159–190, 2000.

[128] Thompson, M. L. and Kramer, M. A., "Modeling chemical processes using prior knowledge and neural networks," *AIChE Journal*, vol. 40, pp. 1328–1340, 1994.

[129] Thornton, J., "High-rate thick-film growth," *Annu Rev Mater Sci*, vol. 7, pp. 239–260, 1977.

[130] Topol, A. W., Dunn, K. A., Barth, K. W., Nuesca, G. M., Taylor, B. K., Dovidenko, K., Kaloyeros, A. E., Tuenge, R. T., and King, C. N., "Chemical vapor deposition of ZnS : Mn for thin-film electroluminescent display applications," *Journal of Materials Research*, vol. 19, no. 3, pp. 697–706, 2004.

[131] Tripathi, A. B., *In-Situ Diagnostics for Metalorganic Chemical Vapor Deposition of YBCO.* PhD thesis, California Institute of Technology, 2001.

[132] Tu, R., Kimura, T., and Goto, T., "Rapid synthesis of yttria-partially-stabilized zirconia films by metal-organic chemical vapor deposition," *Materials Transactions*, vol. 43, pp. 2354–2356, 2002.

[133] Tulleken, H. J. A. F., "Grey-box modelling and identification using physical knowledge and bayesian techniques," *Automatica*, vol. 29, no. 2, pp. 285–308, 1993.

[134] van Duyne, R., Hulteen, J., and Treichel, D., "Atomic force microscopy and surface-enhanced Raman spectroscopy. I. Ag island films and Ag film over polymer nanosphere surfaces supported on glass," *J. Chem. Phy.*, vol. 99, pp. 2101–2115, 1993.

[135] Vargas Garcia, J. and Goto, T., "Thermal barrier coatings produced by chemical vapor deposition," *Science and Technology of Advanced Materials*, vol. 4, no. 4, pp. 397–, 2003.

[136] Venables, J. A., Spiller, G., and Hanbuecken, M., "Nucleation and growth of thin films," *Rep. Prog. Phys.*, vol. 47, pp. 399–459, 1984.

[137] von Stockar, U., Valentinotti, S., Marison, I., Cannizzaro, C., and Herwig, C., "Know-how and know-why in biochemical engineering," *Biotechnology Advances*, vol. 21, no. 5, pp. 417–430, 2003.

[138] Wagner, B. and Harvey, J., "Experimental design for estimating parameters of rate-limited mass transfer: Analysis of stream tracer studies," *Water Resources Research*, vol. 33, pp. 1731–1741, 1997.

[139] Wahl, G., Metz, C., and Samoilenkov, S., "Thermal barrier coatings," *Journal De Physique IV*, vol. 11, no. PR3, pp. 835–846, 2001.

[140] Walpole, R. and Myers, R., *Probability and Statistics for Engineers and Scientists*, vol. 5th. New YorkToronto: Maxwell Macmillan International, 1993.

[141] Wang, A. W., "A strategy for modeling of variations due to grain size in polycrystalline thin-film transistors," *IEEE Transactions on Electron Devices*, vol. 47, pp. 1035–1043, 2000.

[142] Watson, I., Atwood, M., Cardwell, D., and Cumberbatch, T., "MOCVD of high-quality $YBa_2Cu_3O_{7-\delta}$ films: *in-situ* preparation of fluorine-free layers from a fluorinated barium source," *Journal of Materials Chemistry*, vol. 4, p. 1393, 1994.

[143] Weiss, L., Amon, C., Finger, S., Miller, E., Romero, D., Verdinelli, I., Walker, L., and Campbell, P., "Bayesian computer-aided experimental design of heterogeneous scaffolds for tissue engineering," *Computer-Aided Design*, vol. 37, no. 11, pp. 1127–1139, 2005.

[144] Wild, C., Koidl, P., Mueller-Sebert, W., Walcher, H., Kohl, R., Herres, N., Locher, R., Samlenski, R., and Brenn, R., "Chemical vapor deposition and characterization of smooth 100-faceted diamond films," *Diamond and Related Materials*, vol. 2, pp. 158–168, 1993.

[145] Wiliams, P. H., "Designing experiments for the modern micro industries," *CEP Magazine*, pp. 58–63, 2006.

[146] Will, J., Mitterdorfer, A., Kleinlogel, C., Perednis, D., and Gauckler, L., "Fabrication of thin electrolytes for second-generation solid oxide fuel cells," *Solid State Ionics*, vol. 131, no. 1-2, pp. 79–96, 2000.

[147] Wold, S. and Sjostrom, M., "Chemometrics, present and future success," *Chemometrics And Intelligent Laboratory Systems*, vol. 44, no. 1-2, pp. 3–14, 1998.

[148] Xiong, R., Wissmann, P., and Gallivan, M., "An extended Kalman Filter for in-situ sensing of yttria-stabilized zirconia in chemical vapor deposition," *Computers & Chemical Engineering*, vol. 30, pp. 1657–1669, 2006.

[149] Xiong, R., *In Situ Sensing for Chemical Vapor Deposition Based on State Estimation Theory.* PhD thesis, Georgia Institute of Technology, School of Chemical & Biomolecular Engineering, 2007.

[150] Xiong, R. and Grover, M. A., "In situ estimation of thin film growth rate, complex refractive index, and roughness during chemical vapor deposition using a modified moving horizon estimator," *Journal Of Applied Physics*, vol. 103, no. 12, 2008.

[151] Xu, Y.-N., Gu, Z.-q., and Ching, W. Y., "Electronic, structural, and optical properties of crystalline yttria," *Phys. Rev. B*, vol. 56, no. 23, pp. 14993–15000, 1997.

[152] Young, T. F., Liu, T. S., Jung, D. J., and Hsi, T. S., "Microstructural and electrical studies of nitrogen doped diamond thin films grown by microwave plasma CVD," *Surface & Coatings Technology*, vol. 200, no. 10, pp. 3145–3150, 2006.

[153] Yu, Q. and Esche, S. K., "A multi-scale approach for microstructure prediction in thermo-mechanical processing of metals," *Journal Of Materials Processing Technology*, vol. 169, no. 3, pp. 493–502, 2005.

[154] Zhang, J. and Adams, J., "FACET: a novel model of simulation and visualization of polycrystalline thin film growth," *Model Simul Mater Sc*, vol. 10, no. 4, pp. 381–401, 2002.

[155] Zhang, J. and Adams, J., "Modeling and visualization of polycrystalline thin film growth," *Computational Materials Science*, vol. 31, no. 3-4, pp. 317–328, 2004.

[156] Zhang, Y., Haynes, J., Pint, B., Wright, I., and Lee, W., "Martensitic transformation in CVD NiAl and (Ni,Pt)Al bond coatings," *Surface & Coatings Technology*, vol. 163, pp. 19–24, 2003.

[157] Zinsmeister, G., "Theory of thin film condensation part b: Solution of the simplified condensation equation," *Thin Solid Films*, vol. 2, pp. 497–507, 1968.