

CHARACTERIZATION OF SUB-10 NM PORES  
IN SILICON-DIOXIDE

by

JOSEPH A. BILLO

Presented to the Faculty of the Graduate School of  
The University of Texas at Arlington in Partial Fulfillment  
of the Requirements  
for the Degree of

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

THE UNIVERSITY OF TEXAS AT ARLINGTON

August 2012

Copyright © by Joseph A. Billo 2012

All Rights Reserved

## ACKNOWLEDGEMENTS

I would like to give my heartfelt gratitude to Dr. Samir Iqbal for starting me down the path of research in semiconductors and bio-nano technology, to Dr. John Priest for teaching me how to solve problems from a manufacturing perspective, and to Dr. Ronald Carter for all his instruction on how to model devices.

I would also like to acknowledge Waseem Asghar, whom without his prior work and instruction I would not have embarked on this goal. Acknowledgements also go to Jacob Jones, Mohammad Hasan, and Nuzhat Mansur who helped with equations and fabrication.

Finally, I would like to thank Amanda Theaker for her unwavering outpouring of love, patience, and support.

May 10, 2012

## ABSTRACT

### CHARACTERIZATION OF SUB-10 NM PORES IN SILICON-DIOXIDE

Joseph Billo, M.S.

The University of Texas at Arlington, 2012

Supervising Professor: Ronald Carter

Solid state nanopores show much promise in the way of functioning as biosensors, But to do so requires that they be manufactured at sizes of 10 nm or less for sensing of proteins. If this is to be so, it is imperative that they be characterized for signal analysis and fabrication.

A method for analyzing the recorded current signals of nanopore protein translocation is presented. The algorithm successfully zero-baselines the unmodified signal to remove discontinuities and sloping and then performs noise-cancelling based on a moving average standard deviation of the signal. It then extracts the amplitude and calculates the widths of each current peak based on threshold detection.

The process by which nanopores are thermally shrunk is thoroughly examined and a mathematical model is derived. This model equation describes the shrinking time of the pore as a function of pore radius for a given temperature. Curve-fitting is performed for this model in order to extract the self-surface diffusion coefficient of  $\text{SiO}_2$ , which is an important parameter needed for the model.

## TABLE OF CONTENTS

ACKNOWLEDGEMENTS .....	iii
ABSTRACT .....	iv
LIST OF ILLUSTRATIONS.....	vii
LIST OF TABLES .....	viii
Chapter	Page
1. INTRODUCTION.....	1
1.1 Types of Nanopores.....	1
1.1.1 Protein Nanopores .....	1
1.1.2 Solid State Nanopores .....	2
1.2 Nanopore Applications .....	2
1.2.1 DNA Sequencing.....	2
1.2.2 Biomarker Detection.....	2
2. NOISE CANCELLING AND PEAK DETECTION OF SENSOR SIGNALS.....	4
2.1 Zero-Baselining of Signal .....	4
2.2 Noise Cancelling of Signal .....	5
2.3 Peak Results .....	6
3. NANOPORE CONSTRUCTION .....	12
3.1 Membrane Formation by Lithography .....	12
3.1.1 Modeling the Etch Process .....	12
3.1.2 Membrane Fabrication Procedure .....	15
3.2 Pore Fabrication .....	20
3.2.1 FIB Drilling.....	20
3.2.2 SEM Shrinking .....	21

3.2.3 Thermal Shrinking .....	21
4. THERMAL SHRINKING MODEL .....	23
4.1 The Volgel-Fulcher-Tammann Equation .....	23
4.2 Surface Free Energy .....	26
4.3 Accounting for Thermal Expansion .....	29
4.4 Negligence of the Loss of Height .....	30
4.5 Deriving the Model .....	31
4.5.1 Derivation Steps .....	31
4.5.2 Application to Previous Data .....	33
4.6 Future Work.....	41
APPENDIX	
A. MATLAB PROGRAM FOR EXTRAPOLATION AND CURVE-FITTING OF EXPERIMENTAL NANOPORE DATA FOR THE THERMAL HEAT SHRINKING MODEL.....	42
B. MATLAB FUNCTION FOR REMOVAL OF NOISE AND EXTRACTION OF PEAKS IN A NANOPORE CURRENT SIGNAL.....	54
REFERENCES .....	76
BIOGRAPHICAL INFORMATION .....	78

## LIST OF ILLUSTRATIONS

Figure	Page
2.1 Zero Baselining of Signal .....	7
2.2 Unmodified Current Signal .....	8
2.3 Zero Baselined Signal .....	9
2.4 Noise Cancelling of Signal .....	10
2.5 Signal After Noise Cancelling.....	11
3.1 Modeling of TMAH Etching .....	13
3.2 Good SiO <sub>2</sub> Membrane Example #1.....	18
3.3 Good SiO <sub>2</sub> Membrane Example #2.....	18
3.4 Cracked SiO <sub>2</sub> Membrane Example #1. ....	19
3.5 Cracked SiO <sub>2</sub> Membrane Example #2 .....	19
4.1 Viscosity vs. Temperature Plot.....	25
4.2 Surface Free Energy Plot.....	28
4.3 Surface Free Energy vs. Pore Radius Plot .....	36
4.4 Mass Flow Rate Per Area Plot .....	36
4.5 Self-Surface Diffusion Coefficient Plot .....	37
4.6 Pore Radius vs. Time Plot.....	38
4.7 1075 °C Pore Depth Comparison.....	39
4.8 1150 °C Pore Depth Comparison.....	40

## LIST OF TABLES

Table	Page
2.1 Detected Peak Data .....	6
3.1 Parameters of Etching Model.....	13
3.2 Required Materials, Chemicals, and Equipment.....	15
4.1 Parameters of VFT Equation.....	24
4.2 Thermally Shrunken Nanopore of Various Average Radii at Two Different Temperatures .....	34



## CHAPTER 1

### INTRODUCTION

Solid state nanopore show a lot of promise for advances in medical technology as biosensors. They can be used to detect for DNA and other biomarkers while being able to be manufacture in existing micro-electromechanical manufacturing facilities. However, the pores must consistently and reliably be manufacture to diameters less than 10 nm. This is due to the size of the biological material that they are being used to detect for. If progress is to be made in making feasible nanopore biosensors, they must be characterized from the positions of signal analysis for material detection, lithography and etching for fabrication, and reliably shrinking the pores to desired sizes below 10 nm.

#### 1.1 Types of Nanopores

##### 1.1.1 Protein Nanopores

Protein nanopores are the oldest type of nanopore, and they are biologically formed with a protein structure embedded in bi-layer of lipids.[1, 2] The most common protein used is alpha-hemolysin, a protein formed from bacteria. This type of nanopore is used due to its size and structure, which is divided into two 5 nm long sections for a total size of a little over 10 nm.[1] Each part of the pore has a different but known molecular structure than the other parts. This allows one to differentiate between materials that pass through the pore.[1]

Unfortunately, protein nanopores have a few undesirable qualities. For one, their biological nature makes them difficult to produce. Because they are made of a protein structure, they will eventually decay and expire. Furthermore, they cannot stand up to extreme changes in salinity, temperature, and PH.[1, 3]

### 1.1.2 Solid State Nanopores

Unlike protein nanopores, solid state nanopores are fabricated into thin film membranes using micro-electromechanical fabrication techniques. SiO<sub>2</sub> is a commonly used film, but silicon-nitride and polymer films have also been used.[4, 5] SiO<sub>2</sub> nanopores are usually drilled as a hole in a membrane of the film using an ion beam, then shrunk down to size. Also unlike protein nanopores, they have been shown to not be very affected by changes in heat, salinity, and temperature.[3]

## 1.2 Nanopore Applications

### 1.2.1 DNA Sequencing

Nanopores show their greatest promise in the field of rapid DNA sequencing. The working method behind this is that DNA has a slight negative charge, and so DNA can be moved with a sufficiently strong enough electric field in a process call electrophoresis.[6] By placing the pore between two halves on an electrolytic solution and applying a voltage bias, the ions in the solution will move through the pore. If DNA is placed in the half with the anode, it too will translocate through the pore but also cause a blocking in doing so. If one monitors the current through the membrane when this happens, the translocation appears as a sharp downward spike in the current.[1, 2, 4, 6-11]

The idea behind rapid DNA sequencing is to correlate these current spikes with the individual base pairs of a strand of DNA. It is thought that, due to their different structures and sizes, different base pairs will produce different values for the peak amplitudes and peak widths.[8, 10, 11] The challenge though is to fabricate nanopores with are approximately 4 nm in diameter, just bigger than the width of a DNA strand.

### 1.2.2 Biomarker Detection

Nanopores are not limited to detection of just DNA. They can detect for many other biological structure by functionalizing them to a specific structure.[1, 9]. Functionalization can be

done with many materials dependent on what one is functionalizing for, but the general idea is to allow for only one specific type of material or molecular structure to pass through the pore. In this way, one can use electrophoresis to yield a positive/negative test result for a selected material.

This has many uses in the form of protein detection.[2, 9] If one wants to test for a specific mutation or other structural change, one could functionalize a nanopore so that only the modified structure can pass through the pore. If translocations are detected in the form of current spikes during electrophoresis, then it confirms a positive test. This combined with the possibility of DNA sequencing shows great promise for advances in medicinal technology. However, much data will have to be collected and analyzed before concrete medical products can be made. This then calls for a way to quickly analyze the current signals recorded and to extract the peak data contained therein.

## CHAPTER 2

### NOISE CANCELLING AND PEAK DETECTION OF SENSOR SIGNALS

In this chapter, the current signal recorded from a SiO<sub>2</sub> pore translocation experiment is analyzed for peaks that correspond to biological material translocating through the pore. An algorithmic model is detailed on how to achieve zero baselining of the signal as well as how to cancel out the ambient noise inherent when measuring on the nano-Ampere scale. The pore was 11,000 nm in diameter with a depth of 300 nm. Human EGFR protein in an electrolytic solution was translocated through the pore. The voltage across the pore was adjusted such that a current bias of 4  $\mu$ V was applied across the wafer die containing the pore. The signal was recorded into a digital text file in ASCII format and delimited with tabs. Loading this text file into the software package MATLAB 2010b allowed conversion of the text into a numerical matrix.

#### 2.1 Zero-Baselining of Signal

The signal must first be flattened with its horizontal axis baselined at zero. This will make it easier for the peaks to be distinguished from noise due to the removal of current biases that make the amplitude distinction between peaks and noise ambiguous. It also allows for the correct recording of peak amplitudes.

To do this requires the use of a sampling window  $W$  cells wide. This sampling window traverses the entire signal in steps of length  $W$  so that no individual steps overlap. At each step, the arithmetic mean of the signal values in the sample window is calculated. This mean is then subtracted from every value in the signal for that step. Figure 2.1 illustrates this process.

This method of zero-baselining works on the following presuppositions:

1. Overall non-flatness of the signal is caused by ambient environmental current noise that biases and distorts the recorded current signal.

2. When recorded as control data without anything translocating through the pores, the ambient current noise has a flat shape that can be centered on the horizontal axis.
3. The ambient current noise, when centered on the horizontal axis, has a mean value of zero.

Since the signal mostly consists of ambient noise, it can be zero-baselined by calculating the mean of each step of the sample window and subtracting it from the corresponding step in the signal. Since there are so few real peaks compared to the number of noise samples, the peaks have little to no effect on the arithmetic mean calculations. So long as the window length is not too big, the effect of traversing the signal in steps removes non-flatness and discontinuities.

Note that the width must be user-specified for each data set to be worked with, and will require some calibration on the part of the user to get the best results. Only through prior experience and forethought of how many peaks are expected and what constitutes a good result can this calibration be done. For this set of data,  $W = 100$  was used. Figure 2.2 and figure 2.3 show a before and after result of this zero-baselining method.

## 2.2 Noise Cancelling of Signal

The signal is now ready to have the noise mostly removed from it and the peaks analyzed. To do noise cancelling, the sampling window is once again used to traverse the entire signal just as before. However, this time it is the standard deviation of the sampling window  $\sigma$  that is important rather than the mean. In the first  $W$  samples of the signal,  $\sigma$  is set as the initial value of a moving average standard deviation  $\sigma_{AVG}$ . Therefore, it is important that no peaks occur in this very first window. At a sampling frequency of 100 kHz and a sampling window 100 cells wide, no translocations should be allowed to occur for the first millisecond of data recording.

After the first window step and all following window steps, the standard deviation is calculated for the sampling window. This standard deviation is then compared to the moving average standard deviation. If the standard deviation exceeds the moving average by a certain user-specified percent threshold, then that signifies the existence of a possible translocation peak. All values of the signal in the window except the one with the largest magnitude (assumed to be the peak) are set to zero. For this set of experimental data, the threshold was set to 25%. If the standard deviation is less than the moving average, then that signifies that there is no possible peak sample in the window. The standard deviation is rolled into the moving average and all signal values in the window are set to zero. This method is illustrated in figure 2.5 and the results can be seen in figure 2.5.

### 2.3 Peak Results

Now that most of the noise has been removed from the signal, the individual peaks can be extracted and analyzed. The peaks are detected and the remaining noise discounted through simple threshold detection; any peak with an amplitude value beyond some threshold is detected as a true translocation peak. For this set of data, the threshold was set to detect any peak less than -600 nA.

In order to determine the width of each detected peak, the original zero baselined signal (before noise cancelling) is needed. A peak is located in the zero baselined signal by matching the sample numbers for when the peak occurs. This is then traced along both the left and right sides until the horizontal axis is crossed. The difference between the horizontal intercept points gives the width of the peak. The peak widths and amplitudes for all detected peaks are recorded and stored (see Table 2.1).

Table 2.1 Detected Peak Data

Peak Number	Amplitude (nA)	Width ( $\mu$ s)
1	-1091.4	70
2	-2138.2	140
3	-2334.1	260

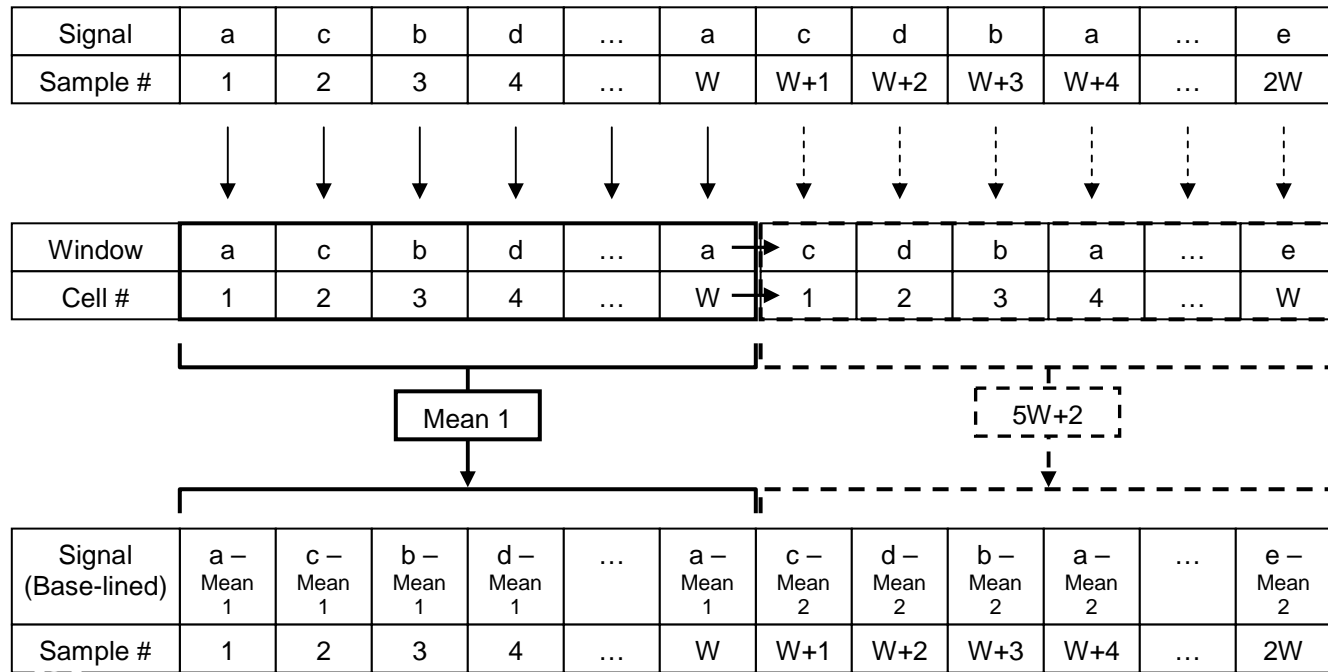


Figure 2.1 A visualization of how a signal is zero-baselined. The noise readings are represented by  $a > b > c > d$ . Peaks are represented by  $e << d$

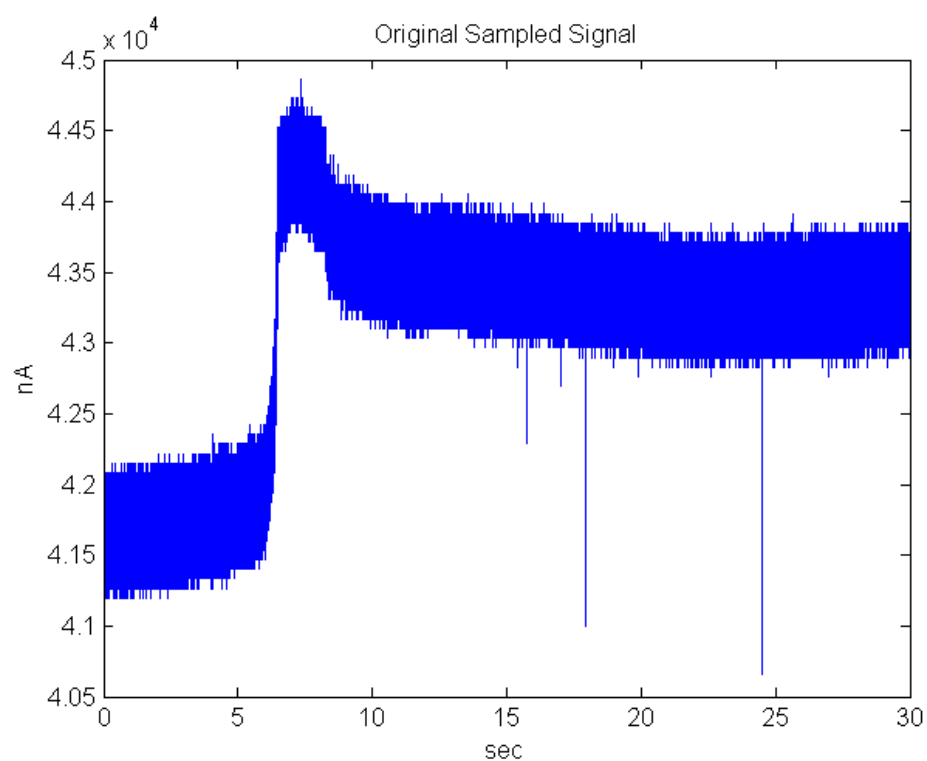


Figure 2.2 The unmodified current signal recorded from human EGFR protein translocating through a 11,000 nm diameter pore in a  $\text{SiO}_2$  membrane at a sampling frequency of 100 kHz for 30 seconds.



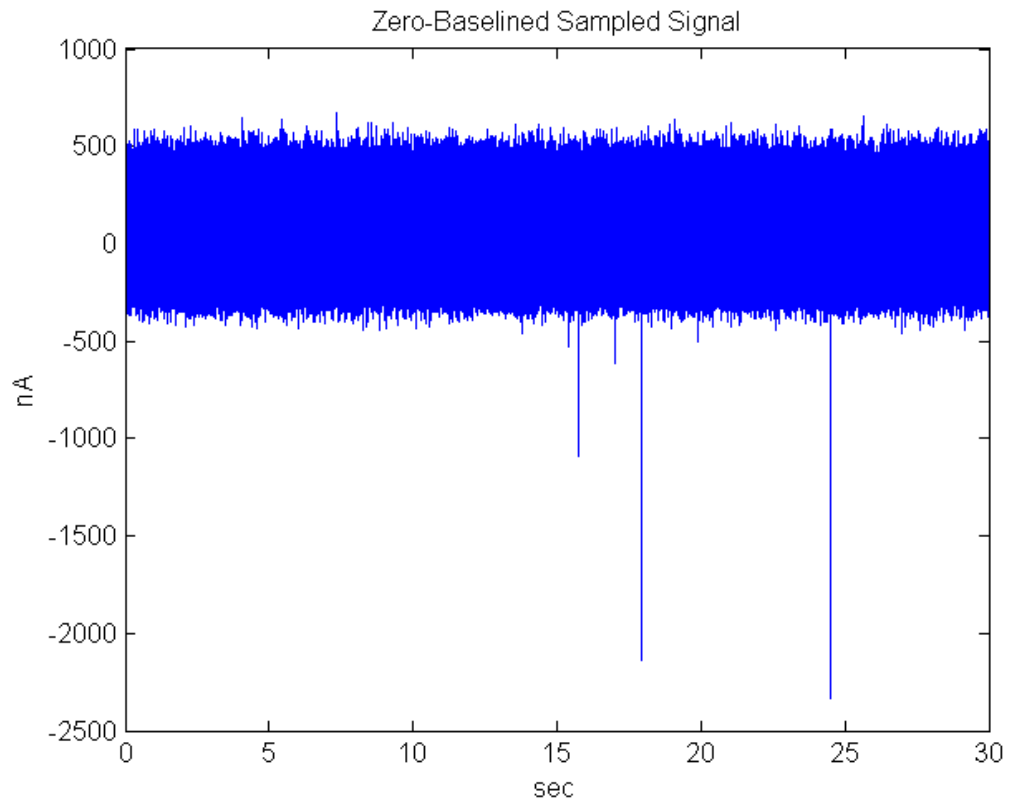


Figure 2.3 The same current signal from figure 2.2, but having undergone zero-baselining with a sampling window of  $W = 100$  cells wide.

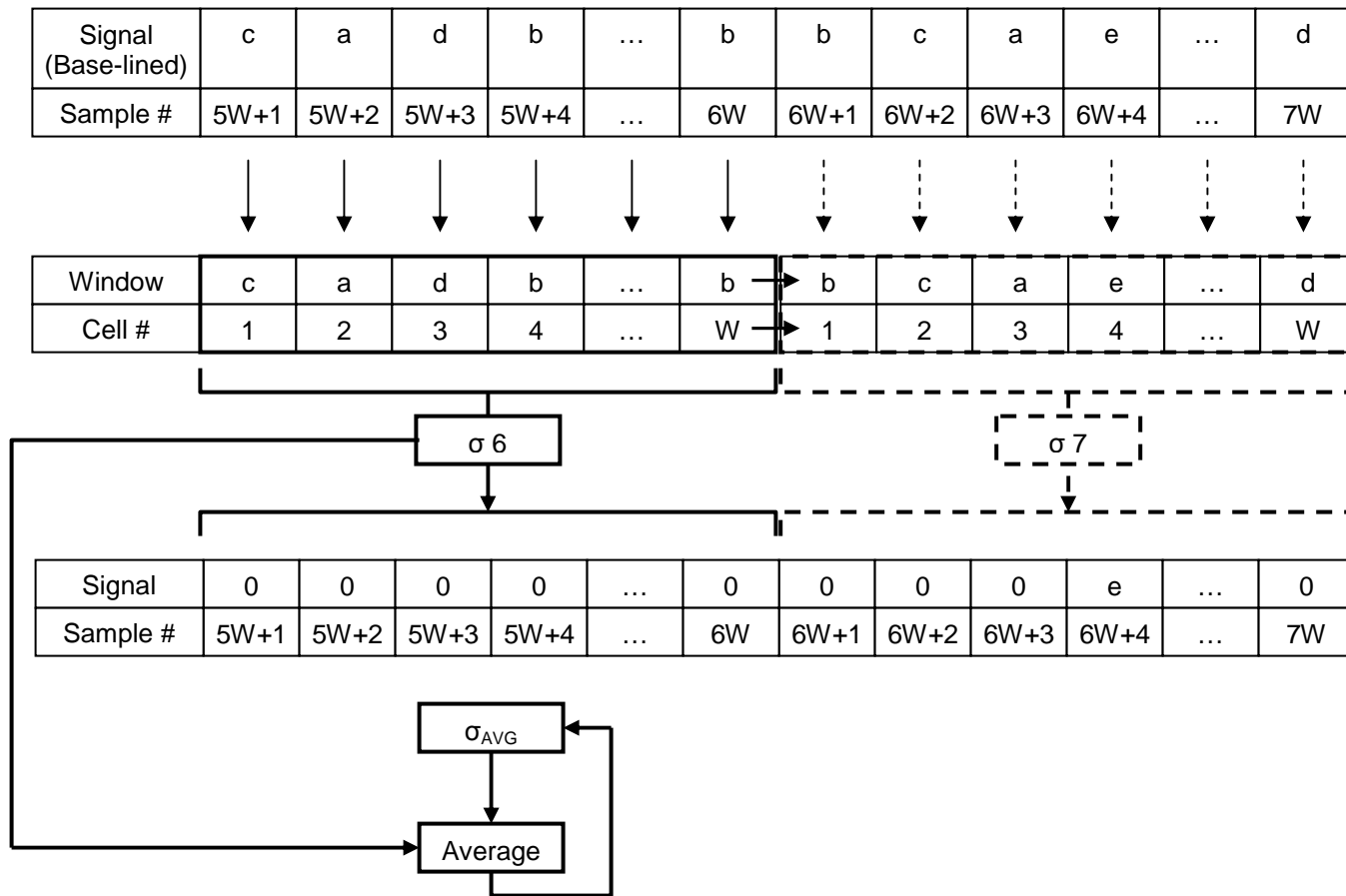


Figure 2.4 A visualization of how a signal has its noise cancelled. The noise readings are represented by  $a > b > c > d$ . Peaks are represented by  $e < c < d$

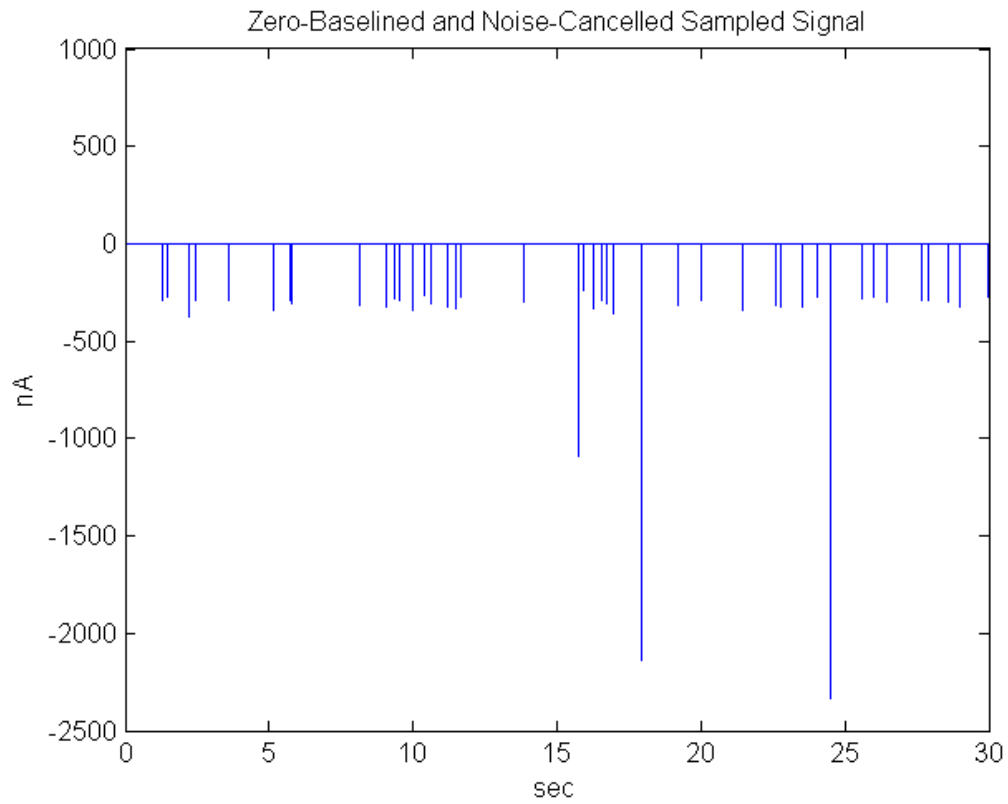


Figure 2.5 The same current signal from figure 2.3, but having undergone noise cancelling with a standard deviation percent threshold of 25%.

## CHAPTER 3

### NANOPORE CONSTRUCTION

Creating a solid-state nanopore is a three-step process. Thin film oxide membranes must first be fabricated on a wafer of silicon. These membranes must then have initial pore drilled that are larger than what is desired. Once this is accomplished, the pores must be shrunk to their appropriate size. In this chapter, a simple geometrical model is created that allows for reliable fabrication of an oxide membrane of a pre-determined size and describes the photolithography steps involved in an experimental fabrication. It then discusses drilling initial pores with a Focused Ion Beam. Finally, two methods of shrinking the pore, electron microspore and thermal shrinking , and discussed.

#### 3.1 Membrane Formation by Lithography

The first step to fabricating solid state nanopores is to first fabricate the membranes into which they shall be drilled. To summarize their fabrication, the first step is to open a square  $\text{SiO}_2$  etch windows in an oxidized silicon wafer. The wafer is then etched anisotropically through its bulk using the back oxide as an etch-stop.[12] Once done, the now-exposed back oxide serves as a square-shaped membrane as thick as the oxide layer into which a nanopore can be drilled[13, 14].

##### 3.1.1 Modeling the Etch Process

For fabricating the membranes, (100) silicon wafers are used. The etchant used to etch the bulk silicon is tetramethylammonium hydroxide (TMAH), which etches along the (111) plane of the wafers. The angle between the (100) and (111) planes is  $54.7^\circ$ . With this information, the etch process can be modeled as follows in Figure 3.1 with the parameters detailed in Table 3.1.

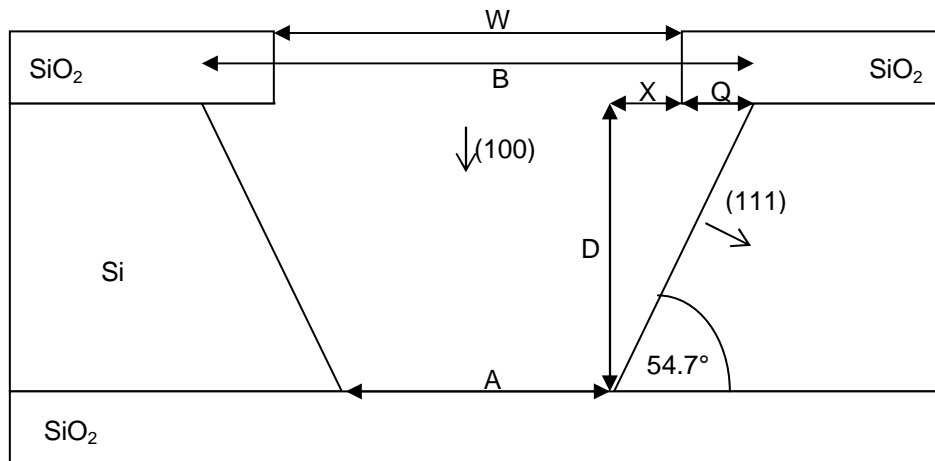


Figure 3.1 A model of how an oxidized silicon wafer is etched in order to create a SiO<sub>2</sub> membrane.

Table 3.1 Parameters of Etching Model

Parameter	Description
D	Thickness of the silicon wafer (excluding oxide)
W	Width of the oxide etch window
A	Width of the membrane
Q	Distance etched underneath the window due to etching in the (111) direction
X	Half the difference in width between W and A
B	The base width of the theoretical trapezoid shape made by the anisotropic etch

This gives three defining equations for the model in Figure 3.1:

$$A = B - 2(Q + X) \quad (3.1)$$

$$W = B - 2Q \quad (3.2)$$

$$\tan(54.7^\circ) = \frac{D}{Q + X} \quad (3.3)$$

It is assumed that  $D$ , and  $Q$  are known ahead of time, and  $A$  is treated as a user-specified parameter. The goal then is to determine how wide the etch window  $W$  must be in order to achieve a membrane the size of  $A$ . This can be done by rearranging equations 3.2 and 3.3 to solve for  $B$  and  $Q+X$  respectively. Substituting these into equation 3.1 and rearranging to solve for  $W$  yields the following equation:

$$A - 2Q + \frac{2D}{\tan(54.7^\circ)} = W \quad (3.4)$$

If  $Q$  is not known ahead of time, it can be calculated using the following equation:

$$Q = \frac{R}{\sin(54.7^\circ)} \quad (3.5)$$

Wherein  $R$  is the distance etched in the (111) direction based on the etch rate of the TMAH solution.

### 3.1.2 Membrane Fabrication Procedure

Table 3.2 Required Materials, Chemicals, and Equipment

1	3" (100) Si wafer pre-oxidized 1 $\mu$ m
2	DI water
3	Methanol
4	Acetone
5	Teflon Wafer Tweezers
6	Stainless Steel Wafer Tweezers
7	Glass Bowl for Developer
8	Teflon Bowl for BHF
9	MF-319 Developer
10	Buffered Hydrofluoric Acid 60% Solution (BHF)
11	Microposit HMDS
12	Shipley S1813 Photoresist
13	Spin-Coater
14	OAI806MBA Backside Aligner
15	Pressurized N <sub>2</sub> Gas
16	Tetramethylammonium Hydroxide (TMAH)
17	4 Liter Capacity Glass Beaker
18	Thermometer
19	Hot Plate with Magnetic Stirring Function
20	Teflon Wafer Holder

The first step in the procedure is to perform a RCA clean on a 3" (100) Si wafer that has been pre-oxidized with 1 $\mu$ m of SiO<sub>2</sub>. The wafer is cleaned with acetone for 10 minutes followed by a de-ionized (DI) water rinse for 5 minutes. Then the wafer is cleaned with methanol for 10 minutes before again being rinsed with DI water for 5 minutes. Finally, a 1:1 Piranha solution is made by adding 50mL of H<sub>2</sub>O<sub>2</sub> to 50mL of H<sub>2</sub>SO<sub>4</sub>. The wafer is bathed in this solution for 10 minutes, and then it is removed and rinsed with DI water for 5 minutes. The wafer is then blown dry with N<sub>2</sub>. In order to remove any remaining moisture from the wafer, it is baked on a hot plate at 200°C for 5 minutes.

The next step is to coat the unpolished side of the wafer with photoresist. In order to make the resist adhere to the unpolished side, it is first coated with Microposit Hexamethyldisilazane (HMDS) Primer. The wafer is placed onto a spin coater and its unpolished surface is covered approximately 50% with HMDS. It is then spun at 3000 rpm for 30 seconds with a 500 rpm/second ramp-up. It is then baked on a hot plate for 90 seconds at

150°C and left to cool. Once cooled, the wafer is then again placed onto the spin coater. This time is coated approximately 2/3rds with Shipley S1813 Photoresist. It is then spun at 2000 rpm for 20 seconds with a 500 rpm/second ramp-up. The wafer then undergoes a soft bake on a hot plate at 115°C for 1 minute and left to cool.

After cooling, the wafer is ready to be exposed and developed. The wafer was loaded into an OAI806MBA Backside Aligner with a mask designed to define the etch windows. For the 3" wafer, the mask opens up square etch windows 500µm on a side. The aligner dosage setting of 20 mW/cm<sup>2</sup> gives an exposure time of approximately 7 seconds. Once exposed, the wafer is developed in MF-319 developer for 35 to 40 seconds with light agitation every 10 seconds. After developing, the wafer is rinsed in DI water for 5 minutes and blown dry with N<sub>2</sub>.

The next step is to coat the polished side of the wafer with photoresist. This side, due to its uniformity, will be the side to contain the membranes. It is coated with photoresist to protect the oxide layer from future steps. Once again it is first coated with HMDS. The wafer is placed onto a spin coater and its polished surface is covered approximately 50% with HMDS. It is then spun at 3000 rpm for 30 seconds with a 500 rpm/second ramp-up. It is then baked on a hot plate for 90 seconds at 150°C and left to cool. Once cooled, the wafer is then again placed onto the spin coater. This time is again coated approximately 2/3rds with S1813. It is then spun at 2000 rpm for 20 seconds with a 500 rpm/second ramp-up. The wafer then undergoes a hard bake on a hot plate at 115°C for 2 minutes and left to cool.

A 60% buffered hydrofluoric acid (BHF) solution is poured into a Teflon bowl at room temperature, and the wafer is submerged in the BHF for 11 to 12 minutes. This isotropically etches the oxide in the exposed and developed sections of the unpolished side of the wafer and creates the etch windows. At the same time, the photoresist coating on the polished side of the wafer keeps the oxide layer that will define the membranes intact and unaffected by the BHF. Once the submersion time has been reached, the wafer is again rinsed with DI water for 5 minutes and blown dry with N<sub>2</sub>.



The wafer is now ready for anisotropic bulk etching with TMAH. The etching solution is prepared by adding 2400 mL of TMAH to 1100 mL of DI water in a 4 L beaker. It is placed atop a magnetic stirring hot plate and the stirrer is dropped into the beaker. The temperature is set to 90°C and the stirring to 200 rpm before the beaker is covered with aluminum foil; a small hole is made in the top for ventilation. Finally, a thermometer is lowered through the ventilation hole in order to verify the temperature of the solution.

While the TMAH solution is heating, another RCA clean is performed on the wafer to remove the photoresist. The wafer is cleaned with acetone for 10 minutes followed by a de-ionized (DI) water rinse for 5 minutes. Then the wafer is cleaned with methanol for 10 minutes before again being rinsed with DI water for 5 minutes. Finally, a 1:1 Piranha solution is made by adding 50mL of  $\text{H}_2\text{O}_2$  to 50mL of  $\text{H}_2\text{SO}_4$ . The wafer is bathed in this solution for 10 minutes, and then it is removed and rinsed with DI water for 5 minutes. The wafer is then blown dry with  $\text{N}_2$ .

When the TMAH has heated to 90°C, the wafer is loaded into a Teflon wafer holder. The wafer is then dipped in BHF for approximately 5 seconds; this is to remove any native oxide layer in the etch windows. The foil covering of the beaker is carefully lifted up and the wafer holder slipped into the beaker. The wafer is then left to etch for 6 hours, checking the temperature every hour or so to keep it at 90°C. After this time has passed, the stirring is reduced to 100 rpm to avoid damage to the forming membranes. After another 1 hour, the wafer is removed from the beaker and is ever so carefully rinsed with DI water for 5 minutes. It is then blown dry with  $\text{N}_2$  using the absolute lowest and gentlest amount of pressure possible. The membranes are now complete and are ready for nanopores to be drilled in them. Such a membrane is shown in Figure 3.2 and Figure 3.3. However, the membranes are fragile. Figures 3.4 and 3.5 are examples of membranes which have been cracked and broken.

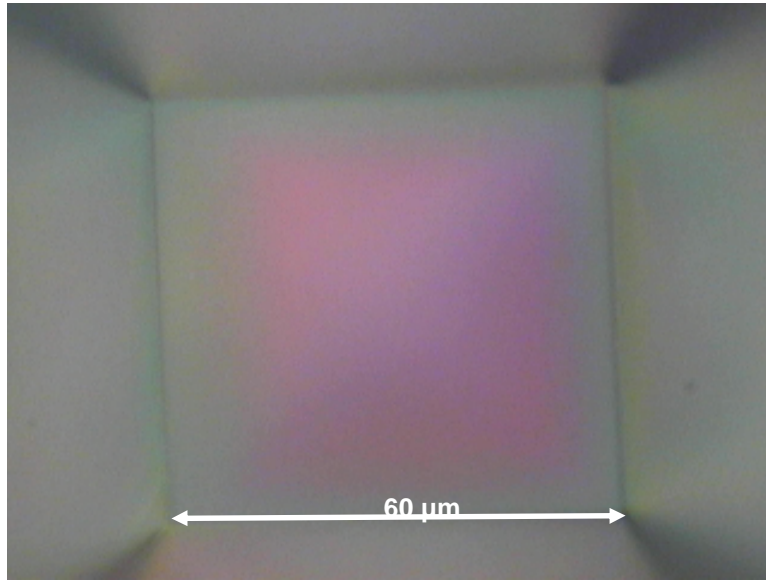


Figure 3.2 A SiO<sub>2</sub> membrane 60 μm wide with a 1 μm thickness after TMAH etching. The point of view is looking down through the etch window.

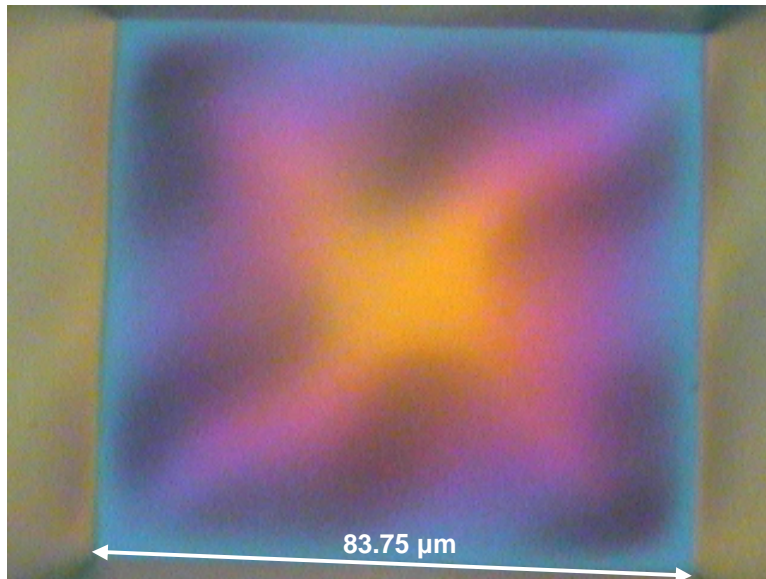


Figure 3.3 A SiO<sub>2</sub> membrane 83.75 μm wide with a 1 μm thickness after TMAH etching. The point of view is looking down through the etch window.

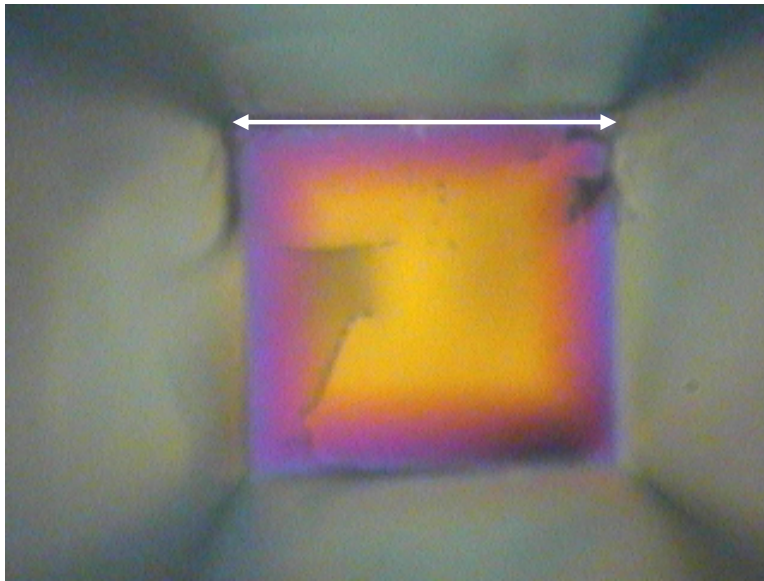


Figure 3.4 A SiO<sub>2</sub> membrane 52.5 μm wide with a 1 μm thickness after TMAH etching. This membrane has been cracked.

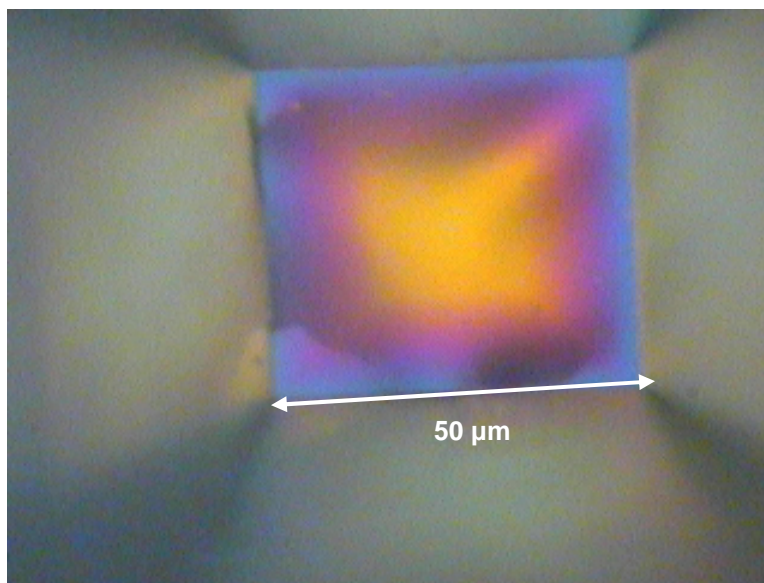


Figure 3.5 A SiO<sub>2</sub> membrane 50 μm wide with a 1 μm thickness after TMAH etching. This membrane has been cracked.

It is important to note that the aluminum foil covering has a unique reaction with TMAH that must be taken into account should the TMAH-filled beaker ever be left idle. The TMAH vapors will react with the aluminum to form aluminum hydroxide.[15] If left to condense, it will form a gel a drip back into the beaker solution. If further left to sit in the beaker, it will crystallize into a solid form that sticks to the inside of the beaker and is not soluble in water. Any acid can be used to remove the aluminum hydroxide in an acid-base reaction. Specifically, using nitric acid will react with the aluminum hydroxide to produce water and aluminum nitrate. To prevent the formation of aluminum hydroxide, it is recommended that a non-reactive covering be used (such as glass) or to replace the aluminum covering approximately every hour during etching.

### 3.2 Pore Fabrication

#### 3.2.1 FIB Drilling

With the membranes complete, fabrication can be continued by opening up the initial pores. To do this, a Focused Ion Beam (FIB) is used to drill a pore directly through each membrane.[12, 16, 17] The beam drills each pore by sputtering ions against the surfaces of the membranes. Though this is a destructive process, it will reliably create initial pores to work with.

Fabricated pores have a target goal diameter of 300 nm.[13] These are fairly large pores that the FIB should be able to easily create. However, it is not possible to get this exact size with every pore drilled. As a compromise, a mean radius of 150 nm is set as the experimental goal when drilling a sample set of membranes. A 30 kV acceleration voltage is used when drilling these pores.[13] The FIB's view screen is used to monitor the sizes of the pores in real time to allow for manual drilling.

Though the pores, in a laboratory setting, are drilled in each membrane one-by-one, it would be possible for FIB drilling to be programmed and implemented in a mass-production setting. This is due to the fact that the membrane thickness, chemical composition, ambient

environment, acceleration voltage, and drilling time can all be controlled ahead of time. By keeping these known parameters constant, drilled pore sizes can be manufactured reliably in pre-determined sizes.[16, 17]

### 3.2.2 SEM Shrinking

Now that the pores are in their membranes, they will have to be shrunk down to size for their appropriate application. For example, pores meant for DNA detection should be slightly bigger than the diameter of DNA, or about 4 nm.[6-8, 13] One way to do this is with a Scanning Electron Microscope (SEM).

Normally, users are warned not to set the current of the SEM too high when taking measurements. Doing so will damage the specimen being observed as it is bombarded by the SEM's electron beam. For shrinking nanopores, this very thing will be done on purpose in order to reduce the pores to their desired sizes. Either a field-emission or transmission SEM can be used. The high current beam, when focused on a pore, damages the perimeter in the SiO<sub>2</sub> membrane. This causes the oxide to flow inward to fill the damaged sections, thereby shrinking the pore in the process.[13, 16] A setting of 300 kV has been used to achieve this, using the SEM's view screen to monitor the sizes of the pores in real time.

While this does allow for accurate pore shrinking, sub-10 nm pore diameters are rather difficult to achieve. Their small size is very hard to image and controllably shrink with the SEM. Furthermore, it has been shown that this process depletes oxygen from the SiO<sub>2</sub> membrane and changes its chemical properties.[13] Finally, this process weakens the structural integrity of the pore. It becomes much easier for the pore to be irreparably damaged after SEM shrinking.

### 3.2.3 Thermal Shrinking

A different method of shrinking is to use thermal heat shrinking to reduce the size of a nanopore.[13] This method involves shrinking the initial pores through the use of a furnace. The

SiO<sub>2</sub> pore membranes are heated to their solidus temperature, causing them to take on a semi-liquid phase. A pore then shrinks as the membrane oxide flows inward towards the pore.[18, 19]

The true solidus temperature of SiO<sub>2</sub> begins somewhere between 800 °C and 900 °C, but no shrinking has been observed over a 20 minute trial at temperatures of 900 °C or less; the ideal temperature range seems to be between 1000 °C and 1200 °C.[13] When pores ranging in size from 100 nm to 300 nm wide were placed in a furnace at these temperatures and in a nitrogen ambient atmosphere, they have been shown to shrink in size proportional to the time spent in the furnace. Sub-10 nm sizes have been achieved using this method as well, and tests have shown that there is no depletion of oxygen.[13]

There are three main advantages that thermal shrinking provides over SEM shrinking. The first is that thermal shrinking involves only a physical change in the pore. No oxygen is depleted, so the inner pore wall retains any chemical and electromagnetic properties. Thermal shrinking can also be used to process an entire sample set of pores by hand in a laboratory setting all at once rather than one-by-one. The pores also strengthened by thermal annealing with this process.[13, 20, 21] With all these advantages over SEM shrinking, the next step would be to model this process so that one could achieve predictability of pore size results.

## CHAPTER 4

### THERMAL SHRINKING MODEL

If thermal heat shrinking is to be a viable process for shrinking SiO<sub>2</sub> nanopores, then it is imperative that one be able to control the process and get predictable results. In this chapter, a mathematical model is created that describes how a pore shrinks with respect to time given a certain temperature. The model takes into account many factors such as viscosity, surface free energy, thermal expansion, dimensional changes due to viscous flow, and self-surface diffusion. Using data from previous works, the model undergoes curve-fitting in order to give an accurate result. Finally, changes in temperature and pore depth are examined for how they change the rate of shrinking.

#### 4.1 The Volgel-Fulcher-Tammann Equation

In order to model how a pore in a SiO<sub>2</sub> behaves during thermal shrinking, one must first have some knowledge about its viscosity at various temperatures. Unfortunately, there is sparse literature for predicting the viscosity of SiO<sub>2</sub> based on a given temperature. It is advantageous then to utilize the Volgel-Fulcher-Tammann (VFT) equation in making such a prediction for a SiO<sub>2</sub> membrane. The VFT is a simple equation for modeling the viscosity of various types of glass at given temperatures.[20] To use it as a model equation for viscosity, a SiO<sub>2</sub> membrane is simply treated as a small-scale adaptation for a pane of 100% amorphous SiO<sub>2</sub> fused-silica glass. The VFT equation is shown as equation 4.1. Its parameters, and their values for pure SiO<sub>2</sub> glass, are listed in table 4.1 [20]

$$\text{Log}_{10}(\eta) = A + \frac{B}{T - T_0} \quad (4.1)$$

Table 4.1 Parameters of VFT Equation

Parameter	Value	Units
$\eta$		dPa-s
$T$		°C
$A$	-7.9250	No Unit
$B$	-31282.9	°C
$T_0$	-415	°C

The parameter  $\eta$  is the calculated viscosity of the glass at temperature  $T$ . This holds true for the model so long as  $T$  is greater than the experimental glass transition temperature.[22] This is particularly advantageous as the VFT equation is accurate in the temperature range for which pores in SiO<sub>2</sub> shrink, which is between 100 and 1200 °C.[13] It is also good to keep in mind that the VFT equation does not model lower temperatures well.[19, 20] Temperatures of a few hundred °C do not curve-fit well, and it is better to use a modified form of the Arrhenius equation instead. Figure 4.1 shows a theoretical plot of viscosity vs. temperature for the given SiO<sub>2</sub> parameters.

Parameters  $A$  and  $B$  are curve-fitting parameters extracted from prior experimental results. The parameter  $T_0$  is the temperature for which configurational entropy disappears in the Gibbs-DiMarzio model for predicting the glass transition temperature.[23] Note that  $T_0$  is simply a theoretical parameter based on the entropy approximations of the Gibbs-DiMarzio theory, and it is not a real temperature one would expect to attain in a laboratory environment. Thus, it is not out of place for this parameter to be lower than absolute zero. Its purpose in the VFT equation is that of a fitting parameter used to account for the inconsistent nature of the activation energy for viscous flow.[19]



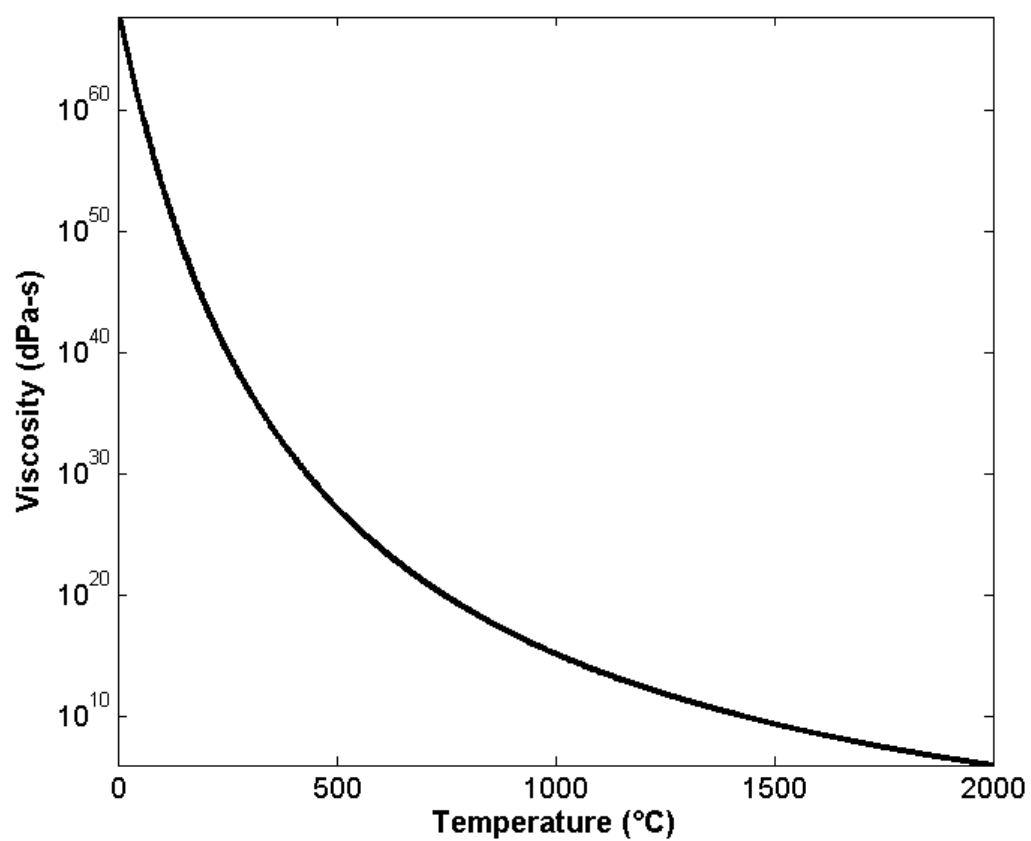


Figure 4.1 Plot of SiO<sub>2</sub> viscosity vs. temperature using the VFT equation and given parameters in table 4.1.

## 4.2 Surface Free Energy

The idea of thermally shrinking a nanopore comes from a phenomenon observed in liquid thin films. If a hole exists in a liquid thin film, it will either shrink or expand in order to minimize the overall surface free energy in the film.[24] By heating up the SiO<sub>2</sub> membrane with a pore in it, the membrane becomes a semi-solid with the same properties as a liquid thin film. As such, the pore will shrink or expand in order to minimize the overall surface free energy.[13, 18]

To determine if the pore will shrink or expand, it is necessary to see how the surface free energy changes with respect to the surface area of the film. The act of shrinking or expanding adjusts the surface area of the film localized at the pore in an attempt to minimize the overall surface tension. This is the mechanism by which the pore changes size. Surface tension is a ratio of the change in surface free energy over an area, and it is proportional to the film's viscosity.[18, 24] Knowing this information results in the following equation:

$$\Delta E = \gamma A \quad (4.2)$$

In equation 4.2,  $\gamma$  is the viscosity of the film (in Pa-s),  $A$  is the surface area, and  $\Delta E$  is the change in surface free energy.

The parameter  $\gamma$  can be obtained via the VFT equation in equation 4.1, albeit with a slight conversion of units needed. For the surface area, the pore is modeled as a cylindrical tube with no top or bottom. Substituting this model in to get the surface area yields:

$$\Delta E = \gamma 2\pi(rh - r^2) \quad (4.3)$$

In equation 4.3,  $r$  is the pore radius and  $h$  is the pore depth or film thickness. With this equation, it is now possible to plot the change in surface free energy with respect to pore radius. A

hypothetical baseline of  $h = 300$  nm,  $T = 1000$  to  $1200$  °C, and  $r = 0$  to  $300$  nm is chosen in order to examine the shape of the plot (Figure 4.2).

Immediately it is seen that the change in surface free energy is zero at both pore radii of  $0$  nm and  $300$  nm. In the instance of  $0$  nm, the pore has shrunk to be completely closed and the surface free energy cannot minimize any more. It is the same in the case of  $300$  nm, but the pore has expanded to its maximum stable size. This is agreeable because equation 4.3 states that  $dE$  is always zero when  $r = 0$  or  $r = h$ .

There is a maximum at  $r = 150$  nm, corresponding to the maximum in equation 4.3 where  $r = 0.5h$ . It is about this maximum that the determination of shrinking or expanding is made. A pore radius to the left of this point will shrink in accordance with the slope. To the right, and the pore will instead expand. Initial pore radius is the only variable in determining which will occur. The maximum itself is theoretically a stable point where no shrinking or expanding will occur. However, it is expected that the slight environmental disturbances in real-world scenarios will cause the pore to lose this stability and either shrink or expand. One can then define a criterion for predicting whether a pore will shrink or expand based on the initial radius of the pore. The pore will shrink if its diameter is less than the film thickness, and it will expand if its diameter is greater than the film thickness.[13, 24, 25]

It is also shown in figure 4.2 that as temperature decreases, there is an exponential increase in the amplitude of the maxima. Thusly, there is an exponential increase in how much change  $dE$  undergoes as it approaches zero. This is due to the fact that lower temperatures will yield a higher viscosity value from the VFT equation, and this will increase all values of  $dE$  in proportion to the increase. It can be extrapolated from this that the surface tension to be overcome in order to shrink the pore also increases exponentially as temperature decreases. This gives a limited window of usable temperatures. Too low and the surface tension will be too great to shrink the pore in a reasonable amount of time. Too high and the pore will be too viscous to shrink it to a predictable size.

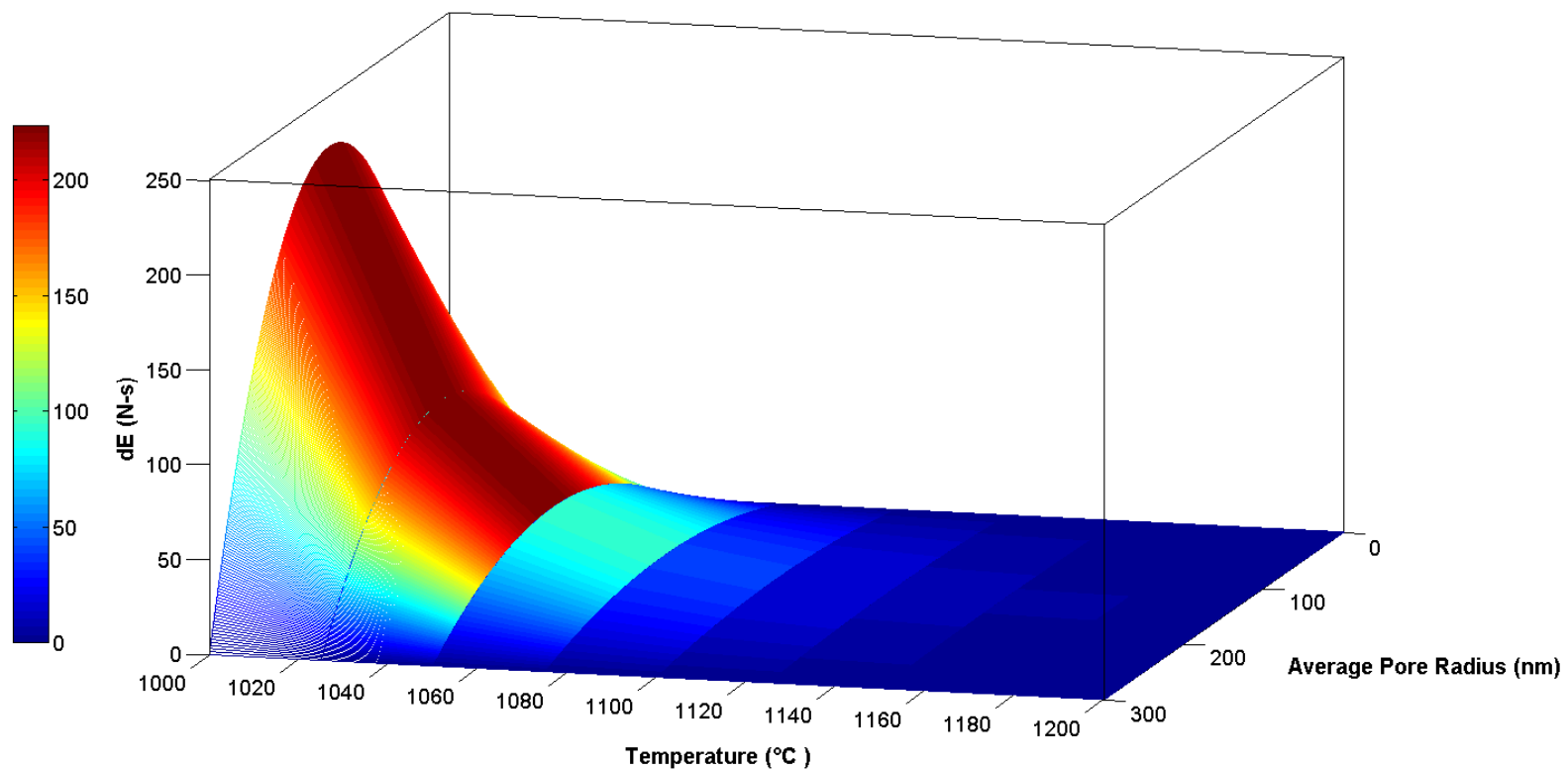


Figure 4.2 Plot of the change in surface free energy  $dE$  with respect to pore radius and temperature. Calculated using equations 4.1 and 4.3 and an assumption of film thickness  $h = 300$  nm.

#### 4.3 Accounting for Thermal Expansion

One parameter that will become important is the number of  $\text{SiO}_2$  molecules per volume are involved in the system of shrinking the nanopore. This number is described by the variable  $N$ . This parameter can be easily calculated from the density,  $d$ , of  $\text{SiO}_2$  at room temperature, but one must account for the fact that the high temperatures involved in shrinking will affect the density through thermal expansion. The parameter  $d_T$ , which is the density at temperature  $T$ , must be calculated. This begins with the thermal expansion equation below:

$$\Delta V = \beta V_0 (\Delta T) \quad (4.4)$$

It is here that  $\Delta V$  is the change in the film's volume,  $V_0$  is the initial film volume,  $\beta$  is the volumetric thermal expansion coefficient for  $\text{SiO}_2$ , and  $\Delta T$  is the change in temperature. This is further broken up into the following:

$$V - V_0 = \beta V_0 (T - T_0) \quad (4.5)$$

It is here that  $T_0$  is room temperature and  $V$  is the volume of the film at temperature  $T$ . The volume parameters are then written as their density equivalents:

$$\frac{m}{d_T} - \frac{m}{d} = \beta \frac{m}{d} (T - T_0) \quad (4.6)$$

Cancelling out the mass  $m$  and solving for  $d_T$  gives the final form of the equation:

$$d_T = \frac{d}{\beta(T - T_0) + 1} \quad (4.7)$$

Given that  $T_0 = 20\text{ }^{\circ}\text{C}$ ,  $\beta$  is approximately  $15 \times 10^{-7}$  per  $^{\circ}\text{C}$ , and  $d$  for  $\text{SiO}_2$  is  $2,648,000\text{ g/m}^3$ ,  $d_T$  can be calculated for any value of  $T$ . This can be taken further in order to calculate the value of  $N$  by dividing  $d_T$  by the molar mass of  $\text{SiO}_2$  then multiplying by Avogadro's number.

#### 4.4 Negligence of the Loss of Height

As the pore shrinks, the  $\text{SiO}_2$  film in which it is made undoubtedly decreases in thickness in order to fill the pore space. This decreases the height  $h$  of the pore which impacts the modeling of the shrinking process. It must be known then if this is something that must be accounted for or can be ignored. To do this requires the volume equations of the film both before and after shrinking. It is assumed that the film has a square shape with a side length  $s$ , a height of  $h$ , and a cylindrical pore in the middle. This yields the following two volume equations for both before shrinking and after shrinking respectively:

$$V_0 = s^2 h_0 - \pi r_0^2 h_0 \quad (4.8)$$

$$V = s^2 h - \pi r^2 h \quad (4.9)$$

It is assumed that both mass of the film and temperature are kept constant during the shrinking process. This allows for equations 4.8 and 4.9 to be set equal to each other:

$$h(s^2 - \pi r^2) = h_0(s^2 - \pi r_0^2) \quad (4.9)$$

Solve for  $h$  in order to find the decreased height after shrinking:

$$h = \frac{h_0(s^2 - \pi r_0^2)}{(s^2 - \pi r^2)} \quad (4.10)$$

Notice the form of this equation. If  $s^2$  is far greater than both  $r_0^2$  and  $r^2$ , then  $h$  will be approximately equal to  $h_0$ .

While solely looking at the SiO<sub>2</sub> membrane in which the pore lies, which may be only tens of microns on one side, it is easy to quickly conclude that yes the height changes significantly. However, to do so is to ignore one vital fact. The oxide of thin film membrane is not isolated to its own borders, but is connected to the oxide of the wafer die in which it is fabricated on. As the membrane loses thickness due to oxide flow, oxide from the die flows in to replace it. The length  $s$  extends far beyond the borders of the membrane, making it many magnitudes of order larger than the  $r$  values. This makes the difference in thickness negligible.

#### 4.5 Deriving the Model

##### 4.5.1 Derivation Steps

With the information from previous sections, it is now possible to derive a full predictive model for thermal nanopore shrinking. As the pore shrinks, the surface tension in the membrane film acts as a normalizing force that resists the shrinking. This normalizing force caused by the surface tension acts as a pressure  $P$  all along the inner surface area of the pore. This can be modeled by the following equation:[25]

$$P = \frac{\Delta E}{(-2\pi r h \Delta r)} \quad (4.11)$$

Notice how equation 4.11 shows the change in surface free energy with respect to the change in radius. This establishes  $P$  as the mass flow rate of the oxide molecules per area, and has units of kg/s per square-meter.[25] With this, equation 4.3 can be substituted in:

$$P = \frac{\gamma 2\pi(rh - r^2)}{(-2\pi rh \Delta r)} \quad (4.12)$$

And this is then further reduced to the following equation:

$$P = \gamma \left( \frac{2}{h} - \frac{1}{r} \right) \quad (4.13)$$

The next needed piece of information is the oxide mobility coefficient  $M$  provided by Lanxner et al.[25] This coefficient is used for determining the flux of the  $\text{SiO}_2$ , and is calculated via the following equation:

$$M = \frac{4(N^{-1/3})^4 D_s}{h^2 kT} \quad (4.14)$$

The two new parameters in equation 4.14 are  $k$ , which is Boltzmann's constant, and  $D_s$ , which is the self-surface diffusion coefficient of  $\text{SiO}_2$ . According to Lanxner et al, multiplying  $P$  and  $M$  gives the rate of change of the pore radius with respect to the rate of change in time (equation 4.15).

$$\frac{dr}{dt} = PM = \left[ \frac{2}{h} - \frac{1}{r} \right] \frac{4(N^{-1/3})^4 D_s \gamma}{h^2 kT} \quad (4.15)$$



This can be rearranged into an equation that can be integrated (equation 4.16). With the knowledge of the initial pore radius  $r_o$ , the desired pore radius  $r$ , and initial  $t = 0$  seconds, this can be integrated over  $r$  to get a general equation relating pore size to time (equation 4.17).

$$\frac{h^2 kT}{4(N^{-1/3})^4 D_s \gamma} \times \frac{1}{\left[ \frac{2}{h} - \frac{1}{r} \right]} dr = dt \quad (4.16)$$

$$t = \frac{h^3 kT}{16 D_s \gamma N^{\frac{4}{3}}} \left[ 2(r - r_o) + h \log_e \left( \frac{|2r - h|}{|2r_o - h|} \right) \right] \quad (4.17)$$

#### 4.5.2 Application to Previous Data

The self-surface diffusion coefficient of  $\text{SiO}_2$  is the last parameter for which information is needed before equation 4.17 can be used to make predictions of pore size based on time. The only way to obtain this information is to use existing experimental results of thermal pore shrinking trials in order to extract the value of  $D_s$ . By placing all the experimental results along with the other known parameters into equation 4.17 and solving for  $D_s$ , a theoretical curve fit can be made in order to get accurate theoretical values of  $D_s$  for different pore radii.

Previous work by Asghar et al[13] have yielded experimental results of thermally shrunken nanopores. These pores were made in  $\text{SiO}_2$  membranes 300 nm thick and were shrunk in a furnace with a nitrogen ambient at four different temperatures: 900 °C, 1075 °C, 1150 °C, and 1250 °C. The 900 °C temperature produced no noticeable shrinkage over the time they were in the furnace. On the other hand, the 1250 °C temperature produced excessive thermal stresses that shrank the pore far too quickly to be controlled and risked destroying the membranes. The results of the 1075 °C and 1150 °C temperatures are shown in table 4.2 below.

Table 4.2 Thermally Shrunk Nanopore of Various Average Radii at Two Different Temperatures[13]

Temperature (°C)	Furnace Time (s)	Final Average Radius (nm)
1075	0	115
1075	300	90
1075	600	52.5
1075	900	17.5
1075	1020	0
1150	0	127
1150	300	75
1150	600	10
1150	642	1.5

Using this information from Table 4.2, it is possible to use the VFT equation (equation 4.3) to calculate the viscosity for the two given temperatures. The values are  $1.176 \times 10^{14}$  Pa-s at 1075 °C and  $1.159 \times 10^{13}$  Pa-s at 1150 °C. This is then used to plot the change in surface free energy (figure 4.3) and the mass flow rate per area (Figure 4.4) with respect to the pore radius. At sub-10 nm pore radii, the mass flow rate per area decreases very sharply. This indicates that at the pores at these sizes shrink very quickly and are very difficult to control. The fact that the curve for the lower temperature has a much less steep drop-off indicates that lower temperature allows for a slower rate of shrinking and better size control. This also indicates that expansion of the pore, should one have use for such a thing, would be easier to control as the pores increase in radius.

It is believed that  $D_s$  is not a constant value throughout the shrinking process. As the oxide flows, it has a natural inclination to minimize its surface free energy. This has a potential energy gradient associated with it that gives a different value depending on the size of the pore radius. In reality,  $D_s$  is a function of such a gradient. As such, points of its curve must be extracted from Table 4.2 using equation 4.17 ( $r = r_0$  cannot be used). This is shown in Figure 4.5. The results were then curve-fitted with a linear curve fit in order to get a theoretical function of  $D_s$  with respect to pore radius for each temperature. Using the extracted and curve-fitted functions of  $D_s$  with equation 4.17 allows for theoretical curves of pore radius vs. time to be

plotted against the data in Table 4.2. Due to the curve-fitting done with  $D_s$ , these theoretical curves are already fitted (Figure 4.6). The mean-square-errors were calculated for each curve.

There is a noticeable flaw in the mean-square-error for the 1075 °C curve. This is due to the fact that the final data point, where the radius has shrunk to zero, was not used for curve-fitting to  $D_s$ . To do so would have been inappropriate. This is because the pore had already closed prior to observation, and there is no way to know the exact point which it closed.

Figures 4.7 and 4.8 repeat the theoretical curves from Figure 4.6, but they have one major difference. They use hypothetical values for the pore membrane thickness in order to show how the radius vs. time relation is affected by different pore depths. It would seem that equation 4.17 predicts that increasing the pore depth flattens the slopes of the curves and increases the time it takes to shrink the pores. This makes sense because by increasing the pore depths one is also increasing the amount of mass that must flow in order to shrink the pore. Naturally, this will require a higher temperature in order for it to flow at the same rate.

This phenomenon is also shown in Figures 4.5 and 4.6 but with temperature. As the temperature increases, the slopes of the curves become sharper and they shrink must faster. This is primarily due to the decreased viscosity caused by the higher temperature. Decreasing the viscosity reduces the surface tension and creates a much higher rate of flow.

This allows for two methods of control for the rate of pore shrinkage: pore depth and temperature. Increasing the pore depth will slow the rate of shrinking to allow for a more tightly-controlled furnace recipe. It also has the added benefit of making the pore membrane more resilient to damage and cracking. Decreasing the temperature has the same effect with the added advantage of being able to adjust the temperature throughout certain points of the shrinking process. It would be feasible to craft a furnace recipe whereby the temperature ramps down when the pore shrinks near sub-10 nm diameters. However, tradeoff for these is time. Slowing the process too much could result in the pores requiring several hours in order to shrink to their desired sizes.

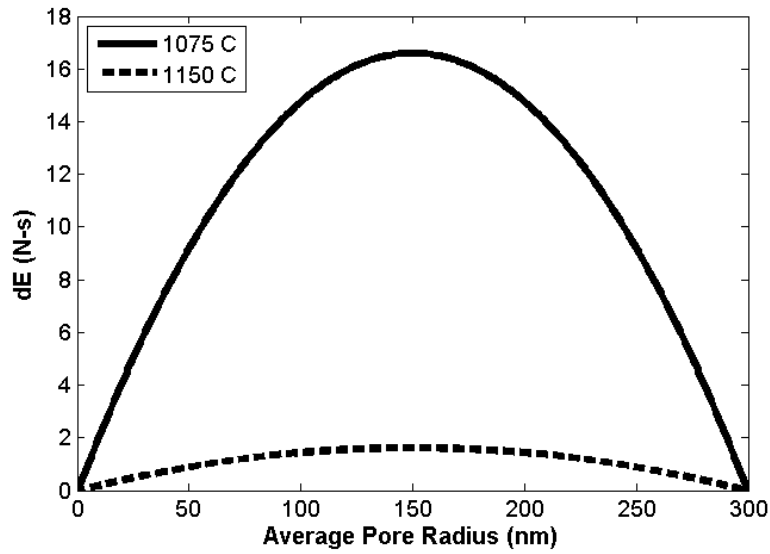


Figure 4.3 Plot of the change in surface free energy  $dE$  with respect to pore radius and temperature for the data in Table 4.2. The viscosity values are  $1.176 \times 10^{-14}$  Pa-s at 1075 °C and  $1.159 \times 10^{-13}$  Pa-s at 1150 °C. The film thickness is  $h = 300$  nm.

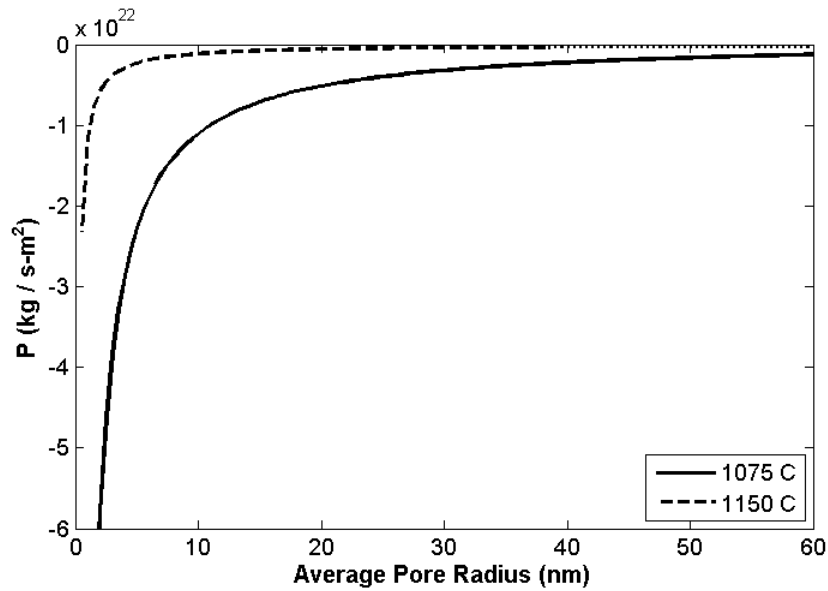


Figure 4.4 Plot of the mass flow rate per area with respect to pore radius.

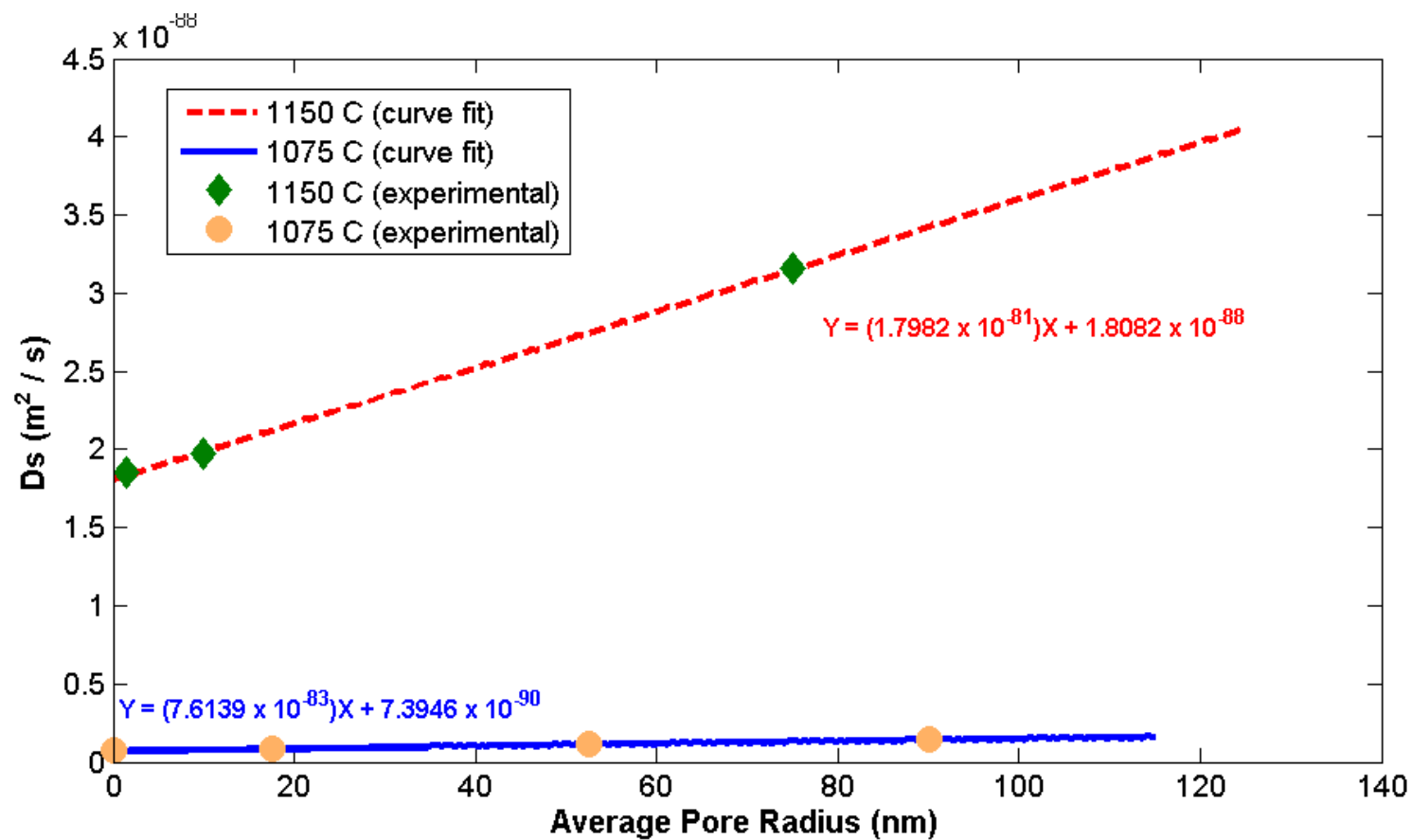


Figure 4.5 Plot of the self-surface diffusion coefficient with respect to average nanopore radius for the 1075 °C and 1150 °C experimental data points. The theoretical curves have been fitted and their linear fits displayed.

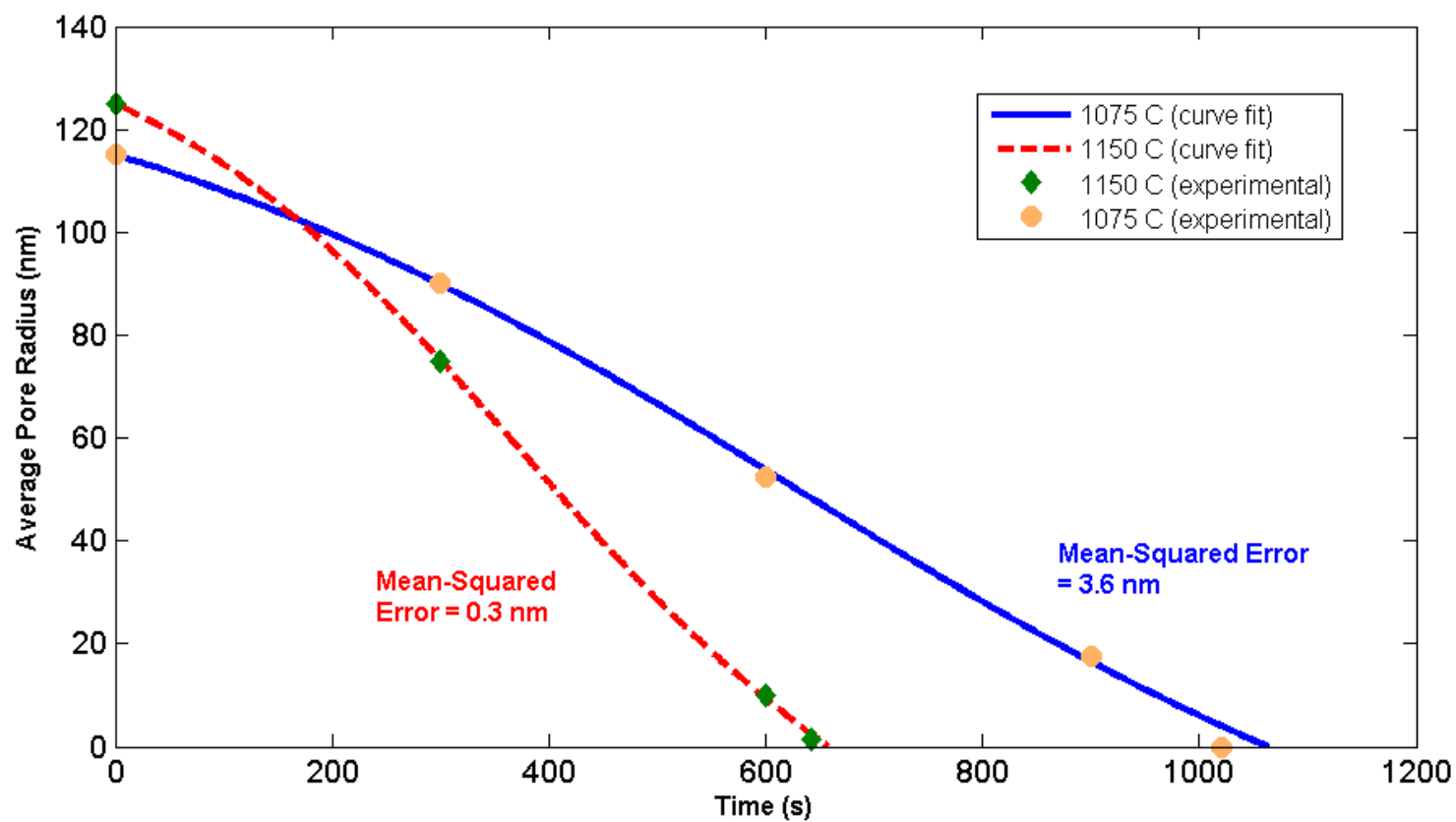


Figure 4.6 Plot of the average pore radius vs. time for the 1075 °C and 1150 °C experimental data points. The theoretical curves have been plotted over them in comparison and their mean-square-errors displayed.

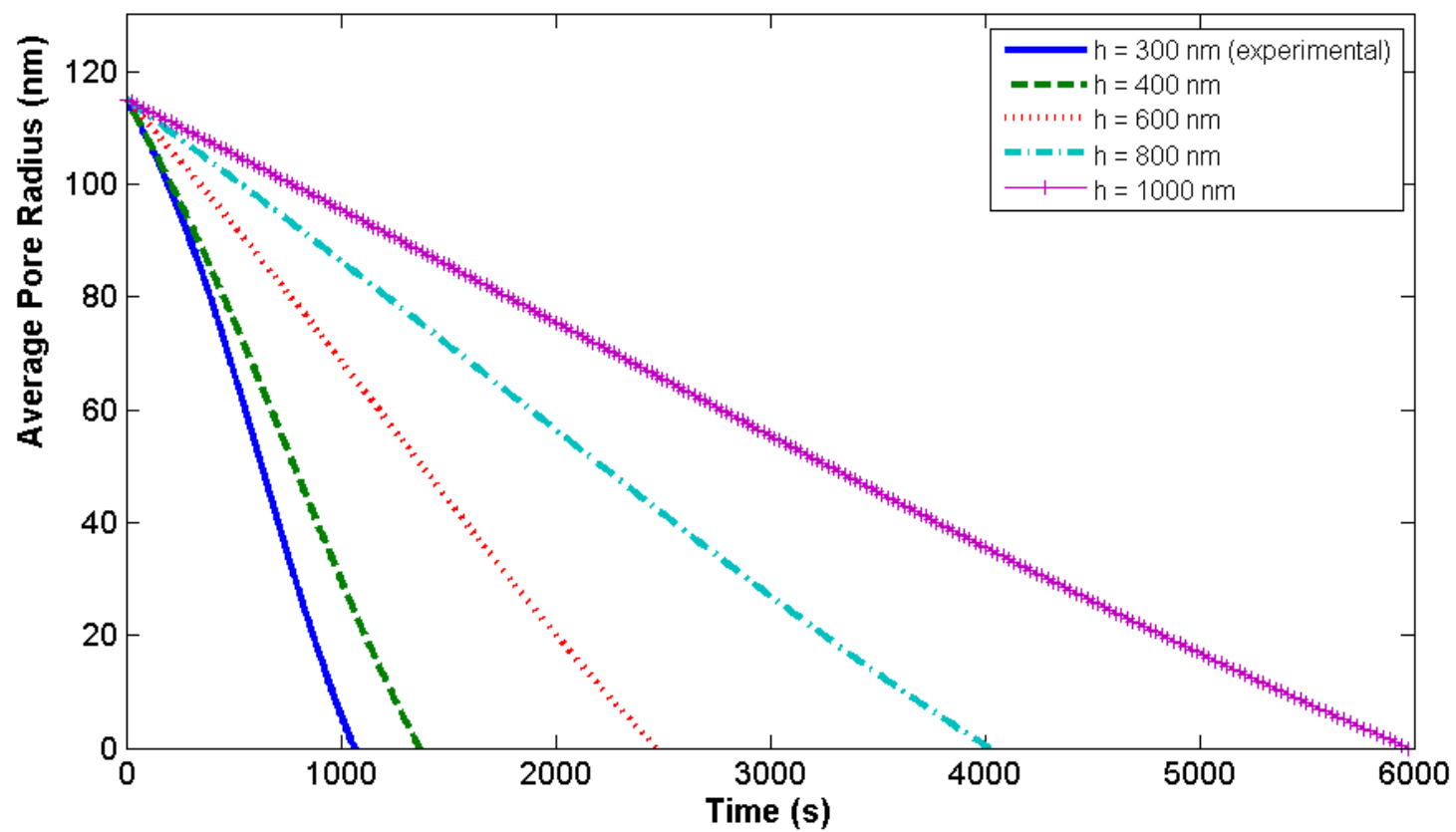


Figure 4.7 Average pore radius vs. time at 1075 °C for several hypothetical pore membrane thicknesses  $h$  alongside the original cure with pore thickness  $h = 300$  nm.

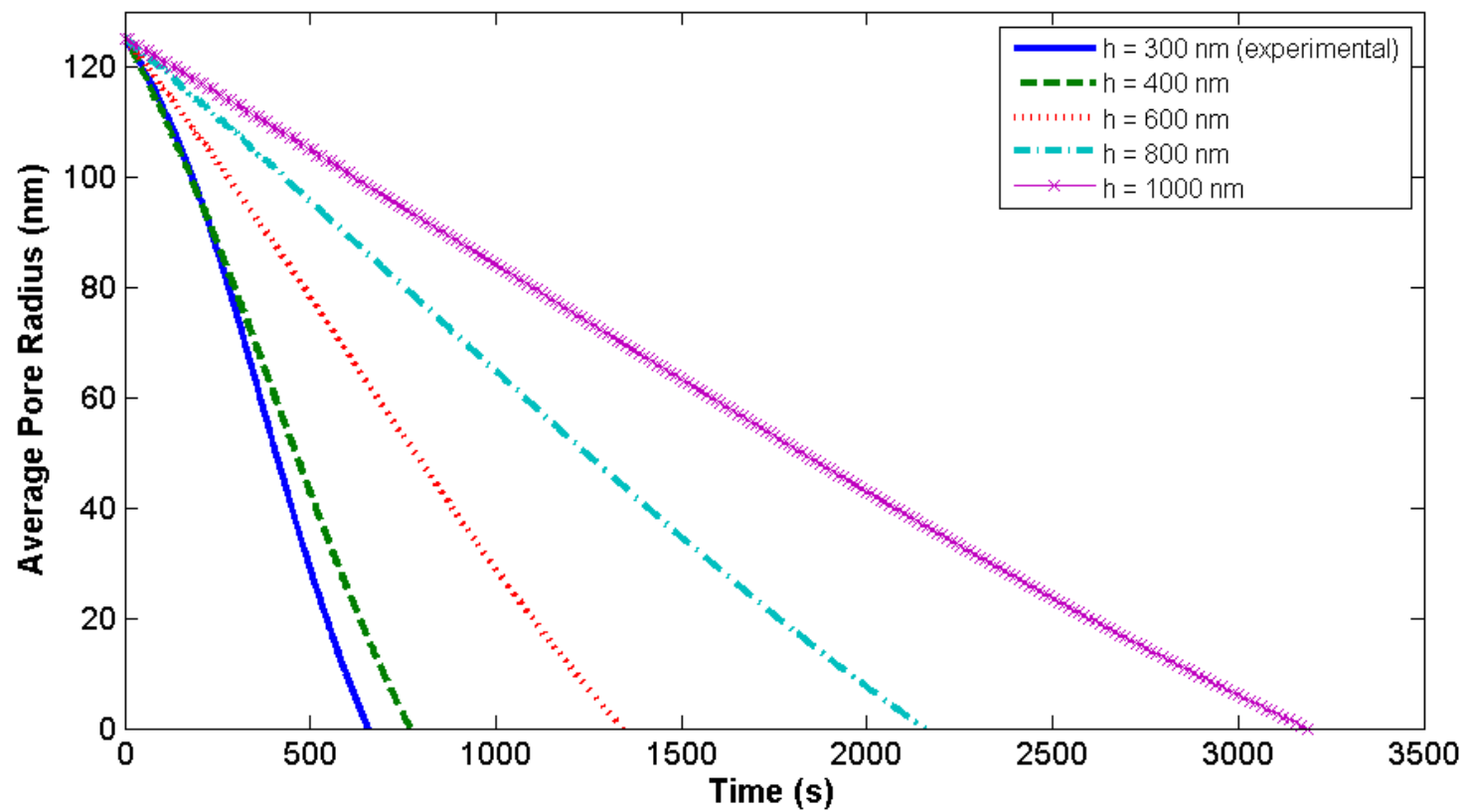


Figure 4.8 Average pore radius vs. time at 1150 °C for several hypothetical pore membrane thicknesses  $h$  alongside the original cure with pore thickness  $h = 300$  nm.



#### 4.6 Future Work

The biggest weakness that must be contended with the thermal shrinking model presented is the requirement to extract new theoretical curves for the self-surface diffusion coefficient. Unfortunately, these will be different for each individual temperature. If one desires to use the model for a process at a certain temperature for which there is no known  $D_s$  data, the only option is to first complete a set of trial experiments at that temperature and extract for oneself the  $D_s$  curve data.

There is the need then for a vast amount of experimental data points on pore shrinking at various temperatures to be accessible. Unfortunately seems to not exist in vast quantities. The primary goal of future work on this model should consist of aggregating theoretical curves of  $D_s$  for reference. This is especially so due to the need for curve-fitting. The more data points there are to fit to, the more accurate the fit. The experimental  $D_s$  curve that's been extracted discretely will always have one less point than the number of data points used to calculate it. Therefore, a minimum of three acceptable pore radius vs. time data points are needed in order to fit a theoretical  $D_s$  curve. This is not encouraged for accuracy reasons, and a minimum of five acceptable data points would be preferable.

Another weakness to contend with is the fact that pinpointing the exact time at which a pore closes completely is incredibly difficult. Visual observation is not possible and most sensing equipment will not be able to withstand the temperatures involved. As a workaround, it would perhaps be better to record data points ever so increasingly nearer to the point where the pore shrinks closed. This way the data points are close enough to zero for an approximation, but one does not risk shrinking them closed without observation.

## APPENDIX A

MATLAB PROGRAM FOR EXTRAPOLATION AND CURVE-FITTING  
OF EXPERIMENTAL NANOPORE DATA FOR THE  
THERMAL HEAT SHRINKING MODEL

%%

% Program: heatshrink4.m

% by: Joseph Billo

% Copyright 2012 University of Texas at Arlington

%

% Description: This MATLAB program extracts the self-surface diffusion

% coefficient and viscosity of SiO<sub>2</sub> from shrunken pore data. It utilizes

% the VFT equation as well as the extrapolated model for pore shrinking.

%%

clear all;

clc;

%Choose data set based on its temperature during shrinking.

dataset = 1075;

deg = 1; %Ds polynomial fit degree

name = int2str(dataset);

%Pure SiO<sub>2</sub> VFT equation parameters.

%Springer handbook of condensed matter and materials data, Volume 1

%By Hans Warlimont

```

A = -7.9250;

B = 31282.9;

T0 = -415.00;

T = dataset;

y = 10 * (10^(A + B/(dataset - T0))) %VFT equation

%Experimental pore shrink data sets
if (dataset == 1150)
    r_exp = [250,150,20,3] * 0.5 * 10^-9
    t_exp = [0,5,10,10.7] * 60
    lastiszero = 0;
elseif (dataset == 1075)
    r_exp = [230,180,105,35,0] * 0.5 * 10^-9
    t_exp = [0,5,10,15,17] * 60
    lastiszero = 1;
end

h = 300 * 10^-9; %Pore depth
N = 2648000 / (60.08*(1.602*10^23)*(1 + (15*10^-7)*(T - 20)));
k = 1.3806488 * 10^-23; %Boltzmann's constant

r0 = r_exp(1);

```

```

figure (1)

plot(t_exp,r_exp,'ro')

title([name, ' C experimental pore data']);

xlabel('time (s)')

ylabel('avg radius (m)')

%Extracting self-surface diffusion coefficient from experimental results

i = 1;

while (i <= length(r_exp))

    ds_exp(i) = ((2*r_exp(i) - 2*r0 + h*log(abs(2*r_exp(i)-h)) - h*log(abs(2*r0-h))) *

((k*T*h^3)/(16*t_exp(i)*N^(-4/3))))/y ;

    i = i + 1;

end

figure (2)

plot(r_exp,ds_exp,'ro')

title([name,' C experimental Ds']);

xlabel('avg pore radius (m)')

ylabel('Ds (m^2 / s)')

%Do curve fitting for Ds

r_th = r0:-0.5*10^-9:0;

temp_r = r_exp;

temp_ds = ds_exp;

```

```

temp_r(1) = []
temp_ds(1) = []
if (lastiszero == 1)
    temp_r(length(temp_r)) = []
    temp_ds(length(temp_ds)) = []
end
temp = polyfit(temp_r, temp_ds, deg);
polyn = temp;
clear 'temp';
clear 'temp_ds';
clear 'temp_r';

%obsolete code
%slope = (ds_exp(3) - ds_exp(2)) / (r_exp(3) - r_exp(2));
%C = ds_exp(2) - slope*r_exp(2);

%Do curve fitting for experimental Ds
i = 1;
while (i <= length(r_th))
    %ds_th(i) = slope*r_th(i) + C; %obsolete%
    ds_th(i) = polyval(polyn, r_th(i));
    i = i + 1;
end
z_radius = r_th;
figure (3)

```

```

plot(r_th*10^9,ds_th,'-',r_exp*10^9,ds_exp,'ro')
title([name, ' C Ds linear fit']);
xlabel('avg pore radius (nm)')
ylabel('Ds (m^2 / s)')

zfig3X1 = r_th*10^9;
zfig3Y1 = ds_th;
zfig3X2 = r_exp*10^9;
zfig3Y2 = ds_exp;

%Use all known parameters to get the theoretical model
i = 1;
while (i <= length(r_th))
    t_th(i) = (2*r_th(i) - 2*r0 + h*log(abs(2*r_th(i)-h)) - h*log(abs(2*r0-h))) *
    ((k*T*h^3)/(16*y*ds_th(i)*N^(-4/3))) ;
    i = i + 1;
end

figure (4)
plot(t_th,r_th*10^9,'-',t_exp,r_exp*10^9,'ro')
title([name, ' C predictive pore fit']);
xlabel('time (s)')
ylabel('avg radius (nm)')

zfig4X1 = t_th;

```

```

zfig4Y1 = r_th*10^9;
zfig4X2 = t_exp;
zfig4Y2 = r_exp*10^9;

%Calculate surface free energy
i = 1;
r_th2 = h:-0.5*10^-9:0; %dE = 0 at r = 0 and r = h
while (i <= length(r_th2))
    dE(i) = y*2*pi*(r_th2(i)*h - (r_th2(i))^2);
    P(i) = ((2/h)-(1/r_th2(i))) * y;
    i = i + 1;
end
r_th2 = r_th2*10^9;
figure (5)
plot(r_th2,dE,'-')
title([name,' C change in surface free energy']);
xlabel('pore radius (nm)');
ylabel('dE (N * s)');
xlim([0, r_th2(1)]);

%Calculate mass flow rate
figure (6)
plot(r_th2,P,'-')
title([name,' C change in mass flow rate per area']);

```



```

xlabel('pore radius (nm)');
ylabel('P (kg/s per m^2)');
xlim([0, r_th2(1)]);

%obsolete

%{
for each r_exp value
    get corresponding ds_exp value
    get cell# of matching r_th value
    use cell# to get corresponding ds_th value

MSE_ds = (ds_th - ds_exp)^2
%}

%Calculate Mean Square Error

i = 2;
while (i <= length(r_exp))
    r = r_exp(i);
    [num,cellnum] = min(abs(r_th-r));
    clear 'num';
    d1 = ds_exp(i);
    d2 = ds_th(cellnum);
    Error(i-1) = (d1-d2)^2;

```

```

    i = i + 1;

end

MSE_ds = mean(Error)

r_exp = r_exp*10^9;
r_th=r_th*10^9;
i = 1;
while (i <= length(t_exp))
    t = t_exp(i);
    [num,cellnum] = min(abs(t_th-t));
    clear 'num';
    r1 = r_exp(i);
    r2 = r_th(cellnum);
    Error2(i) = (r1-r2)^2;
    i = i + 1;
end

MSE_r = mean(Error2)

%zbigarray1 = theoretical radius
%           experimental radius
%           theoretical Ds
%           experimental Ds
%           theoretical time
%           experimental time
i = 1;
while (i <= length(zfig3X1))

```

```

    zbigarray1(1,i) = zfig3X1(i);

    i = i + 1;
end

i = 1;
while (i <= length(zfig3X1))
    a = zfig3X1(i);

    j = 1;

    zbigarray1(2,i) = NaN;
    %zbigarray1(4,i) = NaN;
    zbigarray1(3,i) = zfig3Y1(i);
    while (j <= length(zfig3X2))
        b = zfig3X2(j);

        if(abs(a-b) < 0.1)
            zbigarray1(2,i) = b;
            %zbigarray1(4,i) = zfig3Y2(j)
        end

        j = j + 1;
    end

    i = i + 1;
end

clear 'a';
clear 'b';

i = 1;
j = 1;
while (i <= length(zfig3X1))
    if (zbigarray1(2,i) >= 0)

```

```

        %ans = zbigarray1(2,i)

        zbigarray1(4,i) = zfig3Y2(j);

        j = j + 1;

    else

        %ans = zbigarray1(2,i)

        zbigarray1(4,i) = NaN;

    end

    %ans = zbigarray1(2,i)

    i = i + 1;

end

i = 1;

j = 1;

while (i <= length(zfig3X1))

    zbigarray1(5,i) = zfig4X1(i);

    if (zbigarray1(2,i) >= 0)

        %ans = zbigarray1(2,i)

        zbigarray1(6,i) = zfig4X2(j);

        j = j + 1;

    else

        %ans = zbigarray1(2,i)

        zbigarray1(6,i) = NaN;

    end

    i = i + 1;

end

```

```

zbigarray1 = fliplr(zbigarray1);

z_h = 400 * 10^-9;

i = 1;

while (i <= length(r_th))

    z_time(i) = (2*z_radius(i) - 2*r0 + z_h*log(abs(2*z_radius(i)-z_h)) - z_h*log(abs(2*r0-z_h))) *

    ((k*T*z_h^3)/(16*y*ds_th(i)*N^(-4/3))) ;

    i = i + 1;

end

z_radius = z_radius * 10^9;

trans_rth2 = r_th2.';

trans_dE = dE.';

trans_P = P.';

```

## APPENDIX B

MATLAB FUNCTION FOR REMOVAL OF NOISE AND EXTRACTION OF PEAKS IN A  
NANOPORE CURRENT SIGNAL

# Nanopore Current Signal Noise Filter and Peak Analyzer

V. R-1.1

by Joe Billo

1/31/2011

Made with Matlab R2010a

Copyright 2010, 2011

Department of Electrical Engineering

University of Texas at Arlington

This program will read a raw data file that contains the negative-peak-containing sampled current signal of a nanopore. The file can be in .txt format. It then takes the signal and automatically zero-baselines it. From there, it attempts to remove noise from the signal whilst leaving the peak magnitudes intact. After this, the program will use negative thresholding to detect the peaks. The depth of the peaks, sample numbers, start times, and end times are store in a matrix called "peaks".

peaks:

```
depth(nA) sample# start time(μs) end time(μs)

_____
peak 1 |         |         |         |         |
      |_____ |_____ |_____ |_____ |
peak 2 |         |         |         |         |
      |_____ |_____ |_____ |_____ |
peak 3 |         |         |         |         |
      |_____ |_____ |_____ |_____ |
...

%}
```

% --- Executes on button press in StartButton.

function StartButton\_Callback(hObject, eventdata, handles)

% hObject handle to StartButton (see GCBO)

% eventdata reserved - to be defined in a future version of MATLAB

% handles structure with handles and user data (see GUIDATA)

%clear all;

clc;

STRCurrentFileName = '';

STRCurrentFilePath = '';



```

set(handles.StartButton,'Enable','off');
set(handles.PeakClear,'Enable','off');
set(handles.TabDelimRow,'Enable','off');
set(handles.TabDelimCol,'Enable','off');
set(handles.SampFreq,'Enable','off');
set(handles.WindSize,'Enable','off');
set(handles.StanDev,'Enable','off');
set(handles.PeakThresh,'Enable','off');
set(handles.PeakNumthresh,'Enable','off');
set(handles.autoloadLastFileBox,'Enable','off');
guidata(hObject, handles);
drawnow();

%last file to autoload
autoloadLastFileNum = str2double(get(handles.autoloadLastFileBox,'String'));

%sample size of averaging window (recommend 100)
h = str2double(get(handles.WindSize,'String'));

%filepath to load
[b , a] = uigetfile('.txt');

%!!start the autoloading loop here
autoloadCurrentNum = 0;
autoloadLoopFlag = 1;
autoloadLoopCount = 0;

```

```

while (autoloadLoopFlag == 1)
    %update the file name to grab the next file
    if (autoloadLoopCount >= 1)
        numupdate = str2num(strtok(STRCurrentFileName, '.'));
        autoloadCurrentNum = numupdate + 1;
        numupdate = num2str(autoloadCurrentNum);
        b = [numupdate, '.txt'];
        a = STRCurrentFilePath;
        clear 'numupdate';
    end

    path = strcat(a,b);

    %!check if the path exists. progress to next file if not valid
    while( (exist(path) ~= 2) && (autoloadCurrentNum <= autoloadLastFileNum) )
        numupdate = str2num(strtok(STRCurrentFileName, '.'));
        autoloadCurrentNum = numupdate + 1;
        numupdate = num2str(autoloadCurrentNum);
        b = [numupdate, '.txt'];
        STRCurrentFileName = b;
        a = STRCurrentFilePath;
        clear 'numupdate';
        path = strcat(a,b);
    end

    if (autoloadCurrentNum > autoloadLastFileNum)

```

```

set(handles.StartButton,'Enable','on');
set(handles.PeakClear,'Enable','on');
set(handles.TabDelimRow,'Enable','on');
set(handles.TabDelimCol,'Enable','on');
set(handles.SampFreq,'Enable','on');
set(handles.WindSize,'Enable','on');
set(handles.StanDev,'Enable','on');
set(handles.PeakThresh,'Enable','on');
set(handles.PeakNumthresh,'Enable','on');
set(handles.autoloadLastFileBox,'Enable','on');
guidata(hObject, handles);
drawnow();

beep;

return;

end

```

```

%set file as the last file loaded

set(handles.LastFileTXT,'String',b);
guidata(hObject, handles);
drawnow();

```

```

STRCurrentFileName = b;

STRCurrentFilePath = a;

clear 'a';

clear 'b';

```

```
%path = 'C:\Users\JAB\Documents\MATLAB\test_sample.abf';
```

```
if (size(path) == [1 0])
```

```
    set(handles.StartButton,'Enable','on');
```

```
    set(handles.PeakClear,'Enable','on');
```

```
    set(handles.TabDelimRow,'Enable','on');
```

```
    set(handles.TabDelimCol,'Enable','on');
```

```
    set(handles.SampFreq,'Enable','on');
```

```
    set(handles.WindSize,'Enable','on');
```

```
    set(handles.StanDev,'Enable','on');
```

```
    set(handles.PeakThresh,'Enable','on');
```

```
    set(handles.PeakNumthresh,'Enable','on');
```

```
    set(handles.autoloadLastFileBox,'Enable','on');
```

```
    guidata(hObject, handles);
```

```
    drawnow();
```

```
    beep;
```

```
    return;
```

```
end
```

```
%standard deviation threshold (recommend 1.25)
```

```
stdnum = str2double(get(handles.StanDev,'String'));
```

```
%peak scanning threshold (recommend -600)
```

```
threshold = str2double(get(handles.PeakThresh,'String'));
```

```
%sampling frequency
```

```
Hz = str2double(get(handles.SampFreq,'String'));
```

```

%tab delimited section to import

delimrow = str2double(get(handles.TabDelimRow,'String'));
delimcol = str2double(get(handles.TabDelimCol,'String'));

%status strings
message = "";
percent = 0;

%loading 0 for .abf or 1 for .txt?
extbool = 1;

percheck = 1;
%{
%load the data file
if (extbool == 0)
    y = abfload(path);
    x = 0:1:length(y)-1;
end
%}
if (extbool == 1)

    message = 'Currently loading file';
    set(handles.MssgBox,'String',message);
    guidata(hObject, handles);
    drawnow();

```

```

w = dlmread(path,'\t', delimrow, delimcol);

message = 'File loaded';
set(handles.MssgBox,'String',message);
guidata(hObject, handles);
drawnow();

y = w(:,2);
x = w(:,1);
clear('w');
else

message = 'No file specified.';
set(handles.MssgBox,'String',message);
guidata(hObject, handles);
drawnow();

beep;

return;
end

%plot of raw input signal
figure(1);
plot(x,y);
hold on;

```

```

xlabel('sec');
ylabel('nA');
title('Original Sampled Signal');
hold off;
%figure(1);
%freqz(y);

message = 'Filtering noise.';
set(handles.MssgBox,'String',message);
guidata(hObject, handles);
drawnow();

%calculate initial average standard deviation
i = 0;
while (i < h)
    j(i+1) = y(i+1);
    i = i + 1;
end
standev = std(j);
clear('i');
clear('j');

j=1;
i = h + 0;
z = zeros(1, length(y)); %initialize z array

```

```

while(i <= length(x))

    avg = 0;

    arrChecker = zeros(1,h);

    counter = 0;

    k = 0;

    while(k < h)

        arrChecker(k+1) = y(i-k); %arrChecker holds the sample window

        k = k + 1;

    end

    standev2 = std(arrChecker); %find standard deviation of the window

    %avg holds window average of non-peak values

    if (standev2 > stdnum*standev) %if out of sigma on top side (peaks detected)

        %average only the non-peaks

        k = 0;

        while (k < h)

            if (min(arrChecker) ~= arrChecker(k+1))

                avg = avg + arrChecker(k+1);

                counter = counter + 1;

            end

            k = k + 1;

        end

        avg = avg / counter;

    else %otherwise

        avg = mean(arrChecker); %just get the average of the whole thing

    end

```



```

%attempt to zero the baseline of the output signal

%subtract the average from the z values that correspond to the window.

k = 0;

while(k < h)

    z(i-k) = arrChecker(k+1) - avg;

    y(i-k) = arrChecker(k+1) - avg;

    k = k + 1;

end

y(length(y)) = 0;

z(length(z)) = 0;

```

```

avg = 0;

arrChecker = zeros(1,h);

counter = 0;

k = 0;

while(k < h)

    arrChecker(k+1) = y(i-k); %arrChecker holds the sample window

    k = k + 1;

end

standev2 = std(arrChecker); %find standard deviation of the window

%avg holds window average of non-peak values

```

```

if (standev2 > stdnum*standev)    %if out of sigma on top side (peaks detected)

    %average only the non-peaks

    k = 0;

    while (k < h)

        if (min(arrChecker) ~= arrChecker(k+1))

            avg = avg + arrChecker(k+1);

            counter = counter + 1;

        end

        k = k + 1;

    end

    avg = avg / counter;

else                                %otherwise

    avg = mean(arrChecker);          %just get the average of the whole thing

end

```

```

if (standev2 > stdnum*standev)    %if out of sigma on top side (peaks detected)

    k = 0;

    while (k < h)

        if (min(arrChecker) == arrChecker(k+1))

            z(i-k) = arrChecker(k+1);    %pick the lowest value of the set

        else

            z(i-k) = avg;                %average the rest

        end

        k = k + 1;

    end

else                                %otherwise

```

```

    k = 0;
    while (k < h)
        z(i-k) = avg;          %average as normal
        k = k + 1;
    end

    standev = (standev2+standev)/2; %and readjust standev
end

i = i + h;

percent = 100 * h * j / (length(x)-1);
if (percent >= percheck);
    set(handles.PcntBox,'String',num2str(percent));
    guidata(hObject, handles);
    drawnow();

    percheck = percheck + 1;
end

j = j + 1;
end

percheck = 1;

%transpose output z into coulums and plot
z = transpose(z);

%plot of input signal after baselining

```

```

figure(2);
plot(x,y);
hold on;
yl = ylim;
xlabel('sec');
ylabel('nA');
title('Zero-Baselined Sampled Signal');
hold off;

```

%plot of output after filtering out noise

```

figure(3);
plot(x,z);
hold on;
ylim(yl);
xlabel('sec');
ylabel('nA');
title('Zero-Baselined and Noise-Cancelled Sampled Signal');
hold off;

```

```

message = 'Processing is complete.';
set(handles.MssgBox,'String',message);
guidata(hObject, handles);
drawnow();

```

```

message = 'Now aggregating peaks and corresponding widths.';
set(handles.MssgBox,'String',message);

```

```

guidata(hObject, handles);

drawnow();

peaks = get(handles.PeakTable,'Data');
peaksName = get(handles.PeakTableFilename,'Data');

if (sum(peaks) == 0)
    counter = 0;
    peaks = [0,0,0,0];
    peaksName = {'n/a'};
else
    [pk1,pk2] = size(peaks);
    counter = pk1;
    clear 'pk1';
    clear 'pk2';
end

i = 0;
CurrentPeakCount = 0;
while (i < length(z))
    if(z(i+1) <= threshold)
        counter = counter + 1;
        CurrentPeakCount = CurrentPeakCount + 1;
        peaks(counter,1) = z(i+1); %peak depth
        peaks(counter,2) = i+1; %the sample number the peak is at max depth
    end
    i = i + 1;
end

```

```

k = 0;
while( y(i+1 - k) <= 0)
    k = k + 1;
end
peaks(counter,3) = (i-k) / Hz * 1000000;

k = 0;
while( y(i+1 + k) <= 0)
    k = k + 1;
end
peaks(counter,4) = (i+k) / Hz * 1000000;

peaksName(counter,1) = {STRCurrentFileName};

%delete rows where peaks are accidentally repeated
%counter = counter

testcounter = 1;
while (CurrentPeakCount - testcounter > 0)
    %debugrow = counter - testcounter

    if ( peaks(counter,3) == peaks(counter - testcounter,3) )
        %debugmessage = 'error'
        peaks (counter,:) = [];
        peaksName (counter,:) = [];
    end
end

```

```

        counter = counter - 1;

        testcounter = testcounter - 1;

        CurrentPeakCount = CurrentPeakCount - 1;
    end

    testcounter = testcounter + 1;
end

%Delete row if end time comes before start time
if ( peaks(counter,3) > peaks(counter,4) )
    peaks (counter,:) = [];
    peaksName (counter,:) = [];
    counter = counter - 1;
    CurrentPeakCount = CurrentPeakCount - 1;
end

end

i = i + 1;

percent = 100 * i / length(z);

if (percent >= percheck);
    set(handles.PcntBox,'String',num2str(percent));
    guidata(hObject, handles);
    drawnow();
    percheck = percheck + 1;
end

end
end

```

```

percheck = 1;

%flag peaks if there are too many from a file
%CurrentPeakCount = CurrentPeakCount
%counter = counter

if (CurrentPeakCount >= str2double(get(handles.PeakNumthresh,'String')) )
    while (CurrentPeakCount >= 1)
        peaksName(counter,2) = {'flagged'};
        counter = counter - 1;
        CurrentPeakCount = CurrentPeakCount - 1;
    end
end

set(handles.PeakTable,'Data',peaks);
set(handles.PeakTableFilename,'Data',peaksName);
guidata(hObject, handles);
drawnow();

message = 'Program complete.';
set(handles.MssgBox,'String',message);
guidata(hObject, handles);
drawnow();

%clean up variables to free memory

```



```
clear 'Hz';  
clear 'ans';  
clear 'arrChecker';  
clear 'avg';  
clear 'counter';  
clear 'extbool';  
clear 'i';  
clear 'j';  
clear 'k';  
clear 'message';  
clear 'path';  
clear 'peakbool';  
clear 'percent';  
clear 'standev';  
clear 'standev2';  
clear 'stdnum';  
clear 'threshold';  
clear 'x';  
clear 'delimrow';  
clear 'delimcol';  
clear 'yl';  
clear 'percheck';  
clear 'testcounter';  
clear 'CurrentPeakCount';  
clear 'peaks';  
clear 'peaksName';
```

```

%!!end the autoloading loop here

autoloadLoopFlag = 0;

if ( (autoloadCurrentNum < autoloadLastFileNum) && (get(handles.autoloadCheckBox,'Value')
== 1.0) )

    autoloadLoopFlag = 1;

end

autoloadLoopCount = autoloadLoopCount + 1;

end

clear 'autoloadLoopFlag';

clear 'autoloadLoopCount';

clear 'autoloadLastFileNum';

clear 'autoloadCurrentNum';


set(handles.StartButton,'Enable','on');

set(handles.PeakClear,'Enable','on');

set(handles.TabDelimRow,'Enable','on');

set(handles.TabDelimCol,'Enable','on');

set(handles.SampFreq,'Enable','on');

set(handles.WindSize,'Enable','on');

set(handles.StanDev,'Enable','on');

set(handles.PeakThresh,'Enable','on');

set(handles.PeakNumthresh,'Enable','on');

set(handles.autoloadLastFileBox,'Enable','on');

```

```
guidata(hObject, handles);  
drawnow();  
  
clear 'STRCurrentFileName';  
clear 'STRCurrentFilePath';  
clear 'h';  
beep;  
return;
```

## REFERENCES

- [1] S. M. Iqbal and R. Bashir, *Nanopores: Sensing and Fundamental Biological Interactions*: Springer, 2011.
- [2] B. M. Venkatesan and R. Bashir, "Nanopore sensors for nucleic acid analysis," *Nat Nano*, vol. 6, pp. 615-624, 2011.
- [3] R. M. M. Smeets, *et al.*, "Salt dependence of ion transport and DNA translocation through solid-state nanopores," *Nano Lett*, vol. 6, pp. 89-95, 2006.
- [4] M. Firnkes, *et al.*, "Electrically Facilitated Translocations of Proteins through Silicon Nitride Nanopores: Conjoint and Competitive Action of Diffusion, Electrophoresis, and Electroosmosis," *Nano Letters*, vol. 10, pp. 2162-2167, 2010/06/09 2010.
- [5] S. Wu, *et al.*, "Lithography-Free Formation of Nanopores in Plastic Membranes Using Laser Heating," *Nano Letters*, vol. 6, pp. 2571-2576, 2006/11/01 2006.
- [6] H. Chang, *et al.*, "DNA-Mediated Fluctuations in Ionic Current through Silicon Oxide Nanopore Channels," *Nano Letters*, vol. 4, pp. 1551-1556, 2004/08/01 2004.
- [7] H. Chang, *et al.*, "DNA counterion current and saturation examined by a MEMS-based solid state nanopore sensor," *Biomedical Microdevices*, vol. 8, pp. 263-269, 2006.
- [8] D. Fologea, *et al.*, "Detecting single stranded DNA with a solid state nanopore," *Nano Lett*, vol. 5, pp. 1905-1909, 2005.
- [9] S. M. Iqbal, *et al.*, "Solid-state nanopore channels with DNA selectivity," *Nat Nano*, vol. 2, pp. 243-248, 2007.
- [10] A. Ramachandran, *et al.*, "Characterization of DNA-Nanopore Interactions by Molecular Dynamics," *American Journal of Biomedical Sciences*, vol. 1, pp. 344-351, 2009.
- [11] A. Singer, *et al.*, "Nanopore Based Sequence Specific Detection of Duplex DNA for Genomic Profiling," *Nano Letters*, vol. 10, pp. 738-742, 2010/02/10 2010.
- [12] S. R. Park, *et al.*, "Fabrication of nanopores in silicon chips using feedback chemical etching," *Small*, vol. 3, pp. 116-119, 2007.
- [13] I. A. Asghar W, Billo JA, Iqbal SM., "Shrinking of Solid-state Nanopores by Direct Thermal Heating.," *Nanoscale Res Lett.*, May 4, 2011 2011.
- [14] A. J. Storm, *et al.*, "Fabrication of solid-state nanopores with single-nanometre precision," *Nature materials*, vol. 2, pp. 537-540, 2003.
- [15] G. Yan, *et al.*, "An improved TMAH Si-etching solution without attacking exposed aluminum," *Sensors and Actuators A: Physical*, vol. 89, pp. 135-141, 2001.
- [16] H. Chang, *et al.*, "Fabrication and characterization of solid-state nanopores using a field emission scanning electron microscope," *Applied Physics Letters*, vol. 88, p. 103109, 2006.
- [17] C. J. Lo, *et al.*, "Fabrication of symmetric sub-5 nm nanopores using focused ion and electron beams," *Nanotechnology*, vol. 17, p. 3264, 2006.
- [18] G. I. Taylor and D. H. Michael, "On making holes in a sheet of fluid," *Journal of Fluid Mechanics*, vol. 58, pp. 625-639, 1973.
- [19] J. E. Shelby, *Introduction to Glass Science and Technology*: Royal Society of Chemistry, 2005.
- [20] W. Martienssen and H. Warlimont, *Springer handbook of condensed matter and materials data*: Springer, 2005.
- [21] H. Mehrer, *Diffusion in solids: fundamentals, methods, materials, diffusion-controlled processes*: Springer, 2007.

- [22] L. S. Garca-Coln, *et al.*, "Theoretical basis for the Vogel-Fulcher-Tammann equation," *Physical Review B (Condensed Matter)*, vol. 40, p. 5, 1989.
- [23] U. W. Gedde, *Polymer Physics*: Chapman & Hall, 1995.
- [24] F. Behroozi, "Surface tension in soap films: revisiting a classic demonstration," *European Journal of Physics*, vol. 31, p. L31, 2010.
- [25] M. Lanxner, *et al.*, "Summary Abstract: Evolution of hole shape in {100}, {110}, and {111} monocrystalline thin films of gold," *Journal of Vacuum Science & Technology A: Vacuum, Surfaces, and Films*, vol. 5, pp. 1748-1749, 1987.

## BIOGRAPHICAL INFORMATION

Joseph Billo earned his B.S. in electrical engineering at Southern Methodist University in Dallas, TX in the year of 2009. In January of 2010 he enrolled at the University of Texas at Arlington and joined Dr. Samir Iqbal's Nano Bio Lab. There, he found an interest in semiconductors and nanotechnology. Joseph takes great pride in his research, which he has presented in Applied Physics Letters and at SPIE.