ELASTOMERIC SENSING OF PRESSURE WITH LIQUID METAL AND WIRELESS INDUCTIVE COUPLING

By

JACOB DICK

A thesis submitted to the

Graduate School-New Brunswick

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Master of Science

Graduate Program in Mechanical and Aerospace Engineering

Written under the direction of

Aaron D. Mazzeo

And approved by

_____

_____

_____

_____

New Brunswick, New Jersey

October 2017

ABSTRACT OF THE THESIS

ELASTOMERIC SENSING OF PRESSURE WITH LIQUID METAL AND WIRELESS INDUCTIVE COUPLING

by JACOB DICK

Thesis Director:

Aaron D. Mazzeo

This thesis describes resistance-based soft sensors filled with liquid metal and a methodology for inductive coupling of electromechanical responses without wired connections. By compressing a tube filled with liquid metal, the cross-sectional area changed, and the sensor detected pressure based on the associated change in electrical resistance. The objective of this work is to understand the effects of material choice, geometrical layout, and relative position of coils on the sensitivity of the wireless sensor. A material testing machine compressed a polyvinylchloride (PVC) tube, a fluorosilicone tube, and a silicone tube that connected to a reader through an inductively coupled pair of coils. Relating measured phase to applied loads resulted in measured sensitivities of 270 millidegrees/N with PVC tubing, 78 millidegrees/N for fluorosilicone tubing, and 136 millidegrees/N for silicone tubing. In addition, when loading the PVC sensor at a rate of 1.44 mm/min and then 0.031 mm/min, the hysteresis changed by 85%. Strain sensing through inductive coupling has the potential to lead to future developments in haptics, smart gaskets, and epidermal electronics that will benefit from wireless connectivity.

ACKNOWLEDGEMENTS

I would like to thank my professor, Aaron D. Mazzeo for his continuous support throughout the development of my research. I appreciate his patience, and encouragement through this process. I have learned a plethora of research and professional skills from him and would like to thank him for all his help.

I would also like to thank the rest of my thesis committee, Professor Howon Lee, Patrick V. Hull, and Professor Pelegri, for their insight and comments. Their input helps me to grow and succeed as a researcher and a professional.

Also, I would like to thank Xiyue Zoe, Ben Hogan, Watley Charles, Kenny Li, and Weijen Yi for their help with the thesis process, and their insightful comments on how to direct my research.

Finally, I would like to thank my wife Jessica Bangel and family for their continued support helping me be a successful professional engineer.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

Introduction

Problem

Given the nonlinear response to applied forces in soft material-based systems, there are efforts to embed sensors to monitor large displacements and strains.[1–3] Soft actuators and compliant structures often deform into complex shapes. These shapes experience strains that are much higher than the dynamic range of conventional strain sensors. Soft strain sensors can measure strains greater than 100%, while conventional strain sensors have a dynamic range of 1% strain or less.[4] When loaded at 1% strain, conventional sensors fracture, and the electrical connection is lost. Current techniques for measuring large strain on soft materials typically involve wired connections. However, future applications in soft robotics, prosthetics, or smart seals may require strain measurements in places where it is not convenient or feasible to run wires. It is necessary for these applications for the sensor to be completely inside the components because parts in those fields form complex and continuously changing shapes. One approach that eliminates wired connections is magnetic inductive coupling. By measuring the change in impedance of a mutually coupled passive circuit, engineers can monitor the strain on highly flexible and stretchable parts.

Motivation

Measuring large strains on highly flexible and stretchable interfaces is becoming more important with developments in soft robots,[1,5] wearable electronics,[6] and exoskeletons.[7,8] These highly stretchable materials also have the potential to contribute to the areas of E-textiles, such as smart T-shirts,[9] flexible electronics,[10] epidermal

electronics,[11–13] and flexible foot orthotics.[14] The ability to detect strain wirelessly in flexible and stretchable devices will be a major step in measuring these complex motions and deformations. This technology will make it possible to measure strain through nonconductive materials such as Ecoflex, acrylic, and glass. As a result, users will be able to read strain from embedded devices.

Background

Other papers have described extensive research on how to achieve flexible and stretchable electronic devices. Past designs achieved flexibility through methods such as buckled ultra-thin silicon wafers,[15] ultra-thin semiconductors,[16] and electrically conductive rubber.[17] Another approach is to use liquid conductors inside microfluidic channels made of elastomers. When designing these types of sensors researchers have successfully used mercury, carbon, silver, and eutectic-gallium indium (eGaIn).[18] In this thesis, we use eGaIn as the liquid conductor.

EGaIn is a liquid conductor used in soft sensors because its high surface tension and low viscosity (2.4 mPa S) allow it to mold to unique shapes with minimal change in electrical and mechanical properties.[19] The high surface tension and moldability are a result of an oxide layer that forms on its surfaces exposed to oxygen. The resulting surface tension is as high as approximately 0.6 N/m.[19] While eGaIn is conventionally the conductor, the common encapsulating matrix that researchers use for designing microfluidic soft sensors is polydimethylsiloxane (PDMS).[3]

EGaIn injected into microfluidic channels made from polydimethylsiloxane (PDMS) has unique behaviors when placed under pressure.[3] These sensors have a

minimum detectable pressure on the order of 15 kPa, and a maximum pressure on the order of 40 kPa.[3] Towards the top of the pressure range, there is a built-in hysteresis error. This hysteresis is rate dependent and becomes more apparent at above 40 kPa. In the strain domain, these sensors are linearly repeatable over a calibration curve that is also dependent on temperature. A calibration curve, given by equation 1, shows the dependence of resistance on temperature and strain. Where ΔR is the change in resistance of the sensor, R is the original resistance of the sensor, G is the gauge factor, ϵ is the strain,α is the temperature coefficient, and Θ is the temperature change.[3] In addition to wired strain sensing, wireless strain sensing has also been a growing area of research.[20]

$$\frac{\Delta R}{R} = G\epsilon + \alpha\theta \quad (1)$$

Passive wireless sensors either use inductively coupled circuits, or radio frequency circuits to detect strain. In 2002, John Butler et al. created sensors that operated by deforming an inductor while under strain. This deformation causes a change in resonance frequency of the sensor.[21] Butler used an LC tank circuit which had a bulk capacitor with a fixed value. He made the inductor out of 30 AWG magnet wire, and he embedded it in flexible epoxy. Butler used a gate dip meter to detect the changes in resonant frequency when the inductor deformed.

In 2006, Yi Jia et al. made passive sensors that consist of a planar spiral inductor and an inter-digitated capacitor. Like in Butlers' research, a change in geometry of one of the components changes the resonant frequency of the sensor.[22] Yi's capacitor operated by changing the distance between the capacitor plates or by changing the effective width

of each of the capacitator plates. Other researchers have investigated the use of magnetic soft materials to detect strain.[23]

In 2008, Ee Lim Tan et al. used magnetically soft materials to measure strain. A flexible layer covers a magnetically soft material. A permanent magnet then covers the flexible layer. When the sensor deforms, the magnetic harmonic spectrum of the soft magnet changes.[23] Both a DC excitation coil and an AC excitation coil excited the soft magnet. A detector coil measured the harmonic frequency response.

Passive wireless sensors also have applications in the biomedical industry. The sensors can measure intraocular pressure in glaucoma patients.[24] They can measure this pressure by using an LC circuit that operates at a resonant frequency of 350 MHz. The intraocular pressure causes the plates of the variable capacitor to bend towards each other. This effect changes the capacitance of the variable capacitor, and in turn, changes the resonant frequency of the circuit. This shift in resonant frequency makes it possible to measure the pressure. In a similar paper, Mark Allen embedded a passive wireless pressure sensor inside a dog to measure the pressure inside its arteries.[25]

Recently, Seung Hee Jeong combined passive wireless sensing with soft materials by testing a coil that he made from PDMS and injected with eGaIn. He demonstrated that liquid injected into microfluidic channels could also support a wireless transfer of power.[26] His setup consisted of a soft coil made from a tube injected with eGaIn and a reader coil. The soft coil stretched and the reader coil output an AC signal. The reader then measured the response and calculated the power transfer efficiency. The results

show that an increase in resistance negatively affects power transfer efficiency. In this thesis, we aim to understand the characteristics of a similar setup.

Objective

We aim to understand the fundamental science and physics associated with the wireless inductive sensing of liquid metal-filled pressure sensors based on a resistive element. Also, our objective was to understand the effects that the mechanical loading has on the hysteresis of pressure sensing soft sensors. We are looking to understand how the mechanical loadings affect the wireless coupling and sensitivity. We also aim to show that we can embed these sensors in Ecoflex and obtain comparable results.

Content

This thesis focuses on investigating the characteristics and reliability of wireless eutectic-gallium indium (eGaIn) based soft sensors. The wireless sensing setup uses a pair of coupled inductive coils to read the strain or pressure on the sensing element. The sensing element can be a variable capacitor, inductor, resistor, or a crystal oscillator. In this paper, we focus on resistive based sensors that we made by embedding a liquid conductor (eGaIn) in Ecoflex, fluorosilicone rubber, and PVC.

While we were compressing the sensor, we found the results drifted over time. When we loaded the sensor to above 10 N, the resistance vs. loading curve shifted upward. We hypothesized that this was a result of a growing oxidation layer. The eGaIn oxidation layer can grow through diffusion of oxygen through the rubber material. Also, the oxidation layer can break while we load the sensor, and re-form as fresh eGaIn fills

the cracks that form under loading. We investigated the growth of this oxidation layer and determined if it affects the repeatability of the sensor.

We also investigated the effect of stress-induced creep on these sensors. We hypothesized that the creep in the three host materials, silicone rubber, fluorosilicone rubber, and PVC, can have a larger effect on the hysteresis of the sensor than the oxidation. To establish the effects of the creep on the loading curve, we tested the effects due to the rate of loading.

The research concluded with an investigation into how all the earlier effects that we tested on a wired setup affected the wireless sensitivity of the sensor. Also, we wanted to determine how the distance between the coils affected the sensitivity and the effect of embedding the sensor into Ecoflex.

Experimental Design

Resistive Soft Sensors Filled with Liquid Metals

This work focuses on the wired response of a resistive soft sensor. The sensing element consists of a microfluidic tube that we injected with eutectic-gallium indium. The tube had an outer diameter of 1/32" and an inner diameter of 1/16". The total length of the PVC tube was 6.87 inches, and the initial resistance of the PVC tube was 0.38 ohms. The total length of the fluorosilicone tube was 6.98 inches, and the initial resistance of the fluorosilicone tube was 0.51 ohms. The total length of the silicone tube was 6.86 inches, and the initial resistance of the silicone tube was 0.38 ohms. However, the magnitude of the impedance of the sensor is much higher because the sensor also has an inductive component. Equation 2 gives the DC resistance of a conductor:

$$\boldsymbol{\sigma = \rho \frac{l}{A}} \tag{2}$$

where ρ is the resistivity of eGaIn; $29.4 \cdot 10^{-6}$ Ω*cm.[27] L and A are the lengths and the cross-sectional area of the tube respectively. When we apply pressure to the tube, the cross-sectional area of the tube decreases. This decrease in the cross-sectional area increases the resistive component of the impedance. We show this system in figure 1. When we apply a strain to the tube, the length of tube increases, and the resistive component of the impedance increases. Equation 2 is the basic theory on how all resistive based sensors operate. However, soft sensors operate in a different force regime than the conventional copper strain sensors because the mechanical properties of soft materials differ from solid metal sensors.

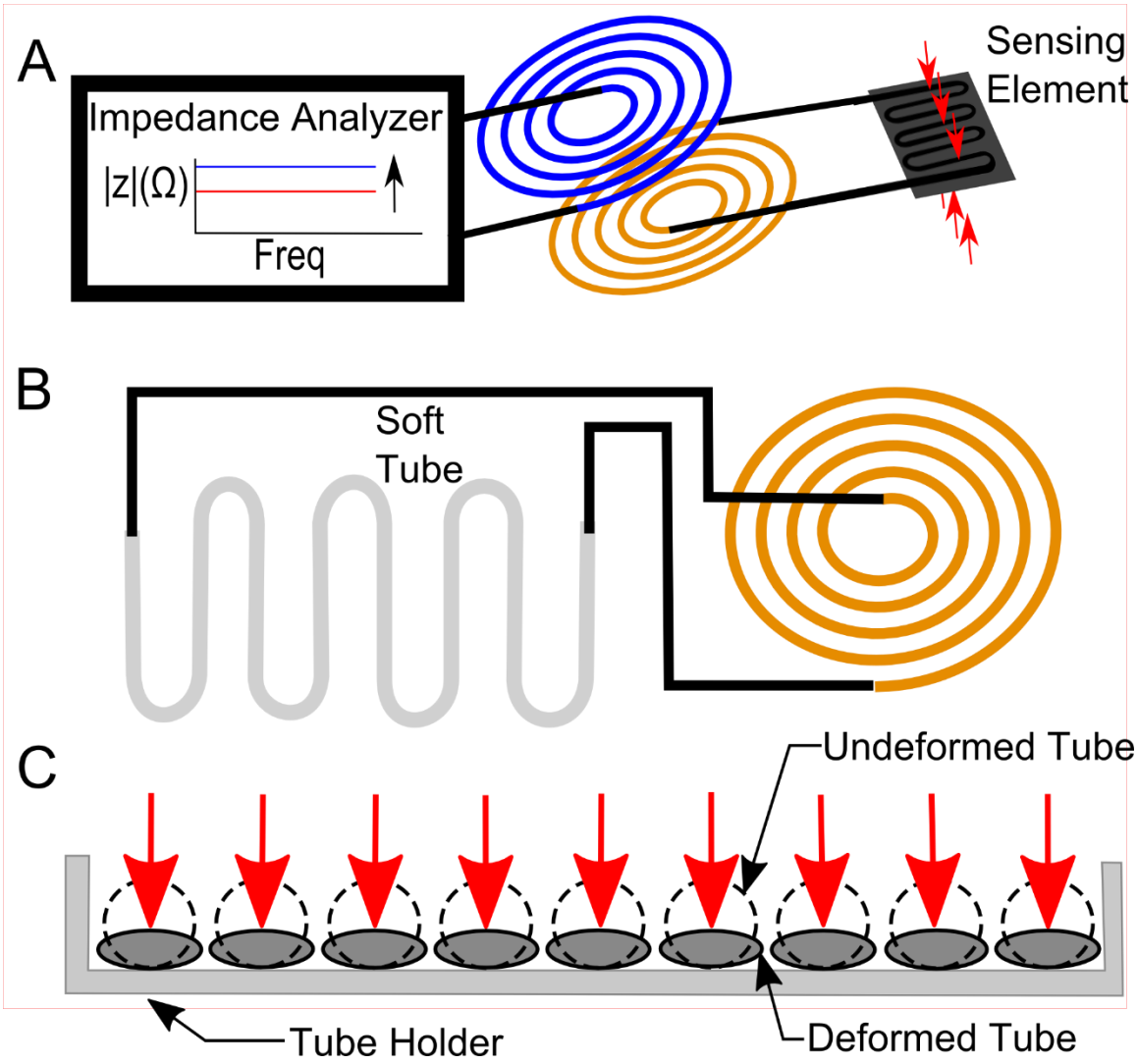

Figure 1: *(a) Overview of the wireless sensing system with a reader and sensor. (b) High-strain, wireless soft sensor. C) Sensor cross section (Credit Ben Hogan).*

The main property that determines the ease by which the sensor can change cross-sectional area is the elastic modulus. Research in the field indicates that we can assume the hydrostatic pressure from the liquid metal is negligible to the applied pressure on the tube.[2] As a result, the elastic modulus of the encapsulating matrix is more important to consider. Most soft materials have moduli of elasticity between 10^2-10^6 Pa. Whereas, most solid metals have moduli of elasticity of 1 GPa or greater.[1] The difference in the elastic moduli of metals when compared to soft materials is three orders of magnitude. This difference allows the soft materials to have applications in soft sensing without as much mechanical impedance. In this paper, we tested three tubes made from PVC, fluorosilicone, and silicone respectively. PVC had an elastic modulus of between $2.5 \cdot 10^9$ Pa and $3.0 \cdot 10^9$Pa.[28] Fluorosilicone rubber had an elastic modulus of $6.9 \cdot 10^7$ Pa.[29] PDMS had an elastic modulus of $7.50 \cdot 10^5$ Pa.[30] The PDMS and the fluorosilicone have lower moduli of elasticity than the PVC.

Elastomers, such as PDMS and flourosilicone, have low elastic moduli because they are made up of cross-linking chains of monomers that have a large number of possible orientations. The high mobility and a large number of orientations for the chains of monomers cause elastomers to have low elastic moduli. By comparison, PVC is a linear polymer and a thermoplastic. The monomer chains of linear polymers coil but do not cross-link with one another. Metallic and ceramic materials require dislocation of molecules or a change in the interatomic distance between the atoms to stretch. As a result, metals and ceramics have a higher elastic modulus than elastomers and PVC. From a thermodynamic perspective, ideal metals experience a change in internal energy

when they undergo stress. Ideal elastomers experience a decrease in entropy when they undergo stress.[31]

We fabricated the sensors using a syringe. Figure 2 illustrates the fabrication process of the sensor. We placed a syringe into one side of the tube and left the opposite end open. This opening allowed oxygen to escape the tube. We then injected eGaIn through the soft tube until the eGaIn was starting to leak out the other end of the tube. We then inserted one copper lead into the open end of the tube. This process assured that the tube was free of excess oxygen, and it caused the eGaIn to be under slight pressure. This pressure causes the eGaIn to rise when we remove the syringe. We then inserted the last copper lead and sealed the tube with hot glue and electrical tape. We used 20 AWG copper wire with 0.2” HMWPE insulation and 1/16” OD diameter tubes. The wire insulation helped to seal the liquid metal inside the tube because the outer diameter of the wire was larger than the inner diameter of the tube. To form a more rigid connection, we used barber tube connectors to join the wires to the tubes. A 3D printed shell held the tube in place while pins compressed the sensor.

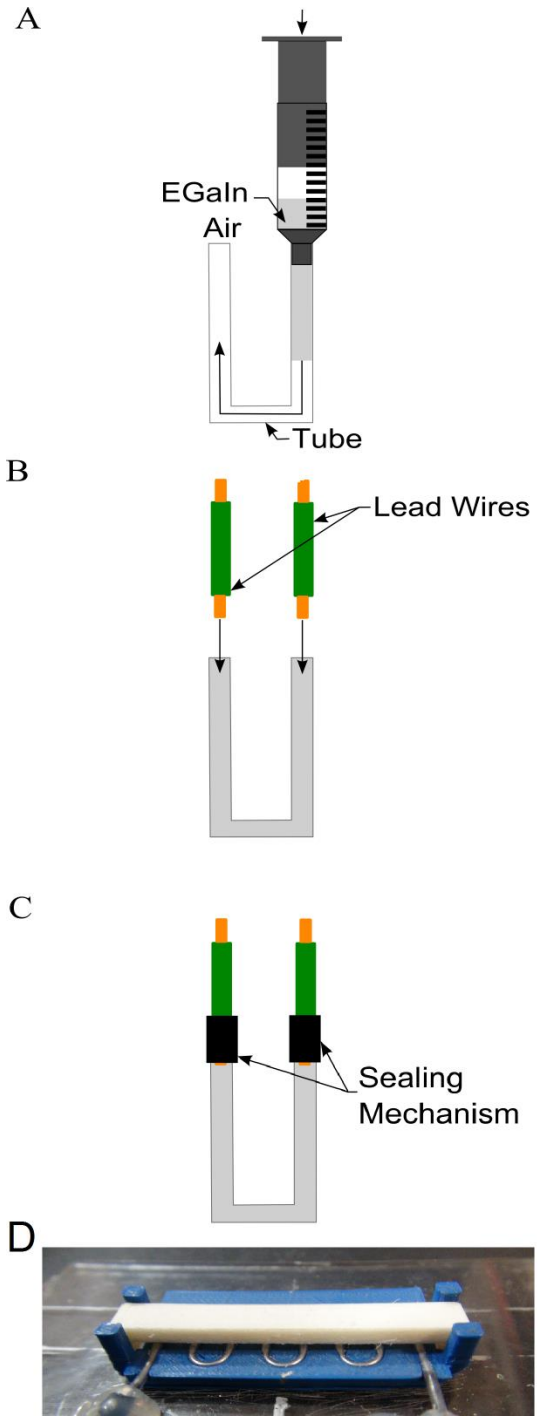

Figure 2: *Fabrication Process of EGaIn filled soft tube. a) A syringe injects EGaIn into the tube. b) We insert leads into to the ends of the tube c) We seal the tubes with hot glue and heat shrink. d) Completed Sensor.*

Inductive Coupling Theory

The inductively coupled sensor consists of four or five components; the reader, the reader inductor, the sensor inductor, a capacitor (optional), and a sensing element. The reader sends out an input signal, and it calculates the impedance of the total system based on the response it receives. The reader inductor mutually will couple to the sensor coil. This mutual inductance between the reader coil and the sensor coil is the mechanism in which the reader communicates with the sensor. Equation 3 gives the effective impedance of the device.

$$\boldsymbol{Z_{eff} = [Z_1 + j\omega L_1 - \frac{(j\omega M)^2}{j\omega L_2 + \frac{1}{j\omega C_2} + R_2 + j\omega L_3}]} \quad (3)$$

The inductors in the system create a tendency for the response to lead the input signal. When we add a capacitor to the sensor side of the system, it creates a tendency for the response to lag the input signal. Those two effects counterbalance each other, and at resonance, they cancel each other out. Resonance is the point of maximum energy transfer between the reader and sensor. This effect is a result of the capacitor releasing the right amount of energy to the inductor at the perfect time to create positive inference between the components. Equation 4 gives the resonant frequency of the device.

$$\boldsymbol{\omega_n = \frac{1}{\sqrt{C_2 L_2}}} \quad (4)$$

The applied strain to the system causes each of the components to deform. The deformation of the capacitor or the inductor causes the resonant frequency to shift. **T**he resistance change causes the phase angle and magnitude of the impedance curves to shift.

Most research on this topic tries to isolate the effects of the deformation to one component. The capacitance of the system changes when we alter the distance between the plates of an interdigitated capacitor.[22] John Butler changed the dimensions of a solenoid to change the impedance curve due to inductance.[21] The impedance due to the resistance of our system changes when we alter the dimensions of the microchannel in our sensor. However, there is a parasitic capacitance associated with each component, and our microchannel had an inductance that caused the imaginary impedance to be much larger than the real component at our operating frequency of 1MHz.

The sensitivity of a non-resonant sensor is dependent on the mutual inductance of the coil pair. Equation 5 is the definition of the mutual inductance of the coil pair. The inductance of the coils is dependent on the geometry of the circuit. The k factor is the quantity that determines the sensitivity of the sensor. We hypothesized in this thesis that the distance between the coil pairs determined the sensitivity of the sensor. Therefore, we believe that there is a relationship between the distance between the coils and the k factor.

$$M = k\sqrt{L_1 L_2} \tag{5}$$

In our experiment, we measured the electromechanical response of the coil pairs with the coil pairs separated ¼" and 1". The medium between the sensors was ambient air. However, we later tested the sensor with acrylic sheets in between the coils. The coils had a width and length of 4 inches, and there were 13 turns. We loaded the sensors at an average rate of 1.25 mm/min. We aimed to observe the change in sensitivity of the sensor based on coil separation.

Electrical Characterization with Impedance Analysis

Our first inquiry into the behavior of the sensor was the chemical oxidation of eGaIn in the microfluidic channels. When eGaIn comes into contact with oxygen, the liquid conductor oxidizes and forms a thin skin made up primarily of GaO_3.[32] This oxidation occurs at the interface between the PDMS and the eGaIn because the PDMS is permeable to oxygen. The oxide layer is not permeable to air, but PDMS is permeable to air. To test our hypothesis, we measured the impedance of an unstrained tube, filled with eGaIn, in an environmental chamber, along with sensors made from various other tube materials, over a three-day period. The environmental chamber kept the temperature and humidity at 25 ˚C and 50% respectively. We tested a silicone rubber tube (McMaster-Carr 5236K203), a fluorosilicone rubber tube (McMaster-Carr 9627T11), and a PVC plastic tube (McMaster-Carr 5233K91). PDMS has a permeability of 60 [cc's (RTP) cm/sec sqcm cm Hg deltaP] with air.[33] Fluorosilicone rubber has a permeability of 11 [cc's (RTP) cm/sec sqcm cm Hg deltaP] with air. PVC has a permeability of 0.014 [cc's (RTP) cm/sec sqcm cm Hg deltaP] with air.

We calculated the RMS difference from the mean by first calculating the mean from each data set. We then subtracted each data point from the mean. We then calculated the RMS value of the resulting data set. To resolve the difference in initial resistance, we normalized the RMS difference from the mean based on the initial resistance. Equation 6 shows how we calculated the RMS difference from the mean. Where $\bar{a}$ is the mean of the data set, and X_i is a point in the data set, N is the number of points in the data set, and R_0 is the intial resistance.

$$RMS_{Mean} = \frac{\sqrt{\frac{(X_1-\bar{a})^2+(X_2-\bar{a})^2+(X_3-\bar{a})^2+\cdots}{N}}}{R_0} \quad (6)$$

Another indication of the variation of the oxidation is the percent drift of the overall resistance. We calculated the percent drift by averaging the data points that the VNA measured in the first 30 minutes of a 72-hour experiment, and then averaging the data points that the VNA measured in the last 30 minutes of the experiment. Equation 7 shows the calculation of the percent drift.

$$Percent\ Drift = \frac{\bar{a}_{Begining}-\bar{a}_{ending}}{\bar{a}_{Begining}} \quad (7)$$

However, we wanted to test whether we could cause the oxide layer to grow by straining the sensor. We believed mechanical strain caused the oxidation layer to break, and fresh eGaIn to move to the surface. This fresh eGaIn layer oxidizes, and the oxide layer grows. We tested this mechanical behavior of the oxide layer by repeatedly compressing the tubes. To perform this test, we attached the three tubes to a material testing machine and compressed them from 0% strain to 75% strain. The testing machine compressed the tubes 15 times. The PDMS experienced a maximum force of 20 N, the fluorosilicone rubber experienced a force of 54 N, and the PVC experienced a force of 53 N. A vector network analyzer measured the behavior of the impedance of the tubes over time. We performed microscopy on samples after repetitive pressure testing to observe the change in the surface of the eGaIn. Figure 3 shows the microscopy image for the PVC. The left side is the uncompressed tube, and the right side is the compressed tube. We noticed that the eGaIn at the core of the tube remains in the same state. However, the

encapsulating matrix shows black bands, which are locations in which the test fixture deformed the matrix material.

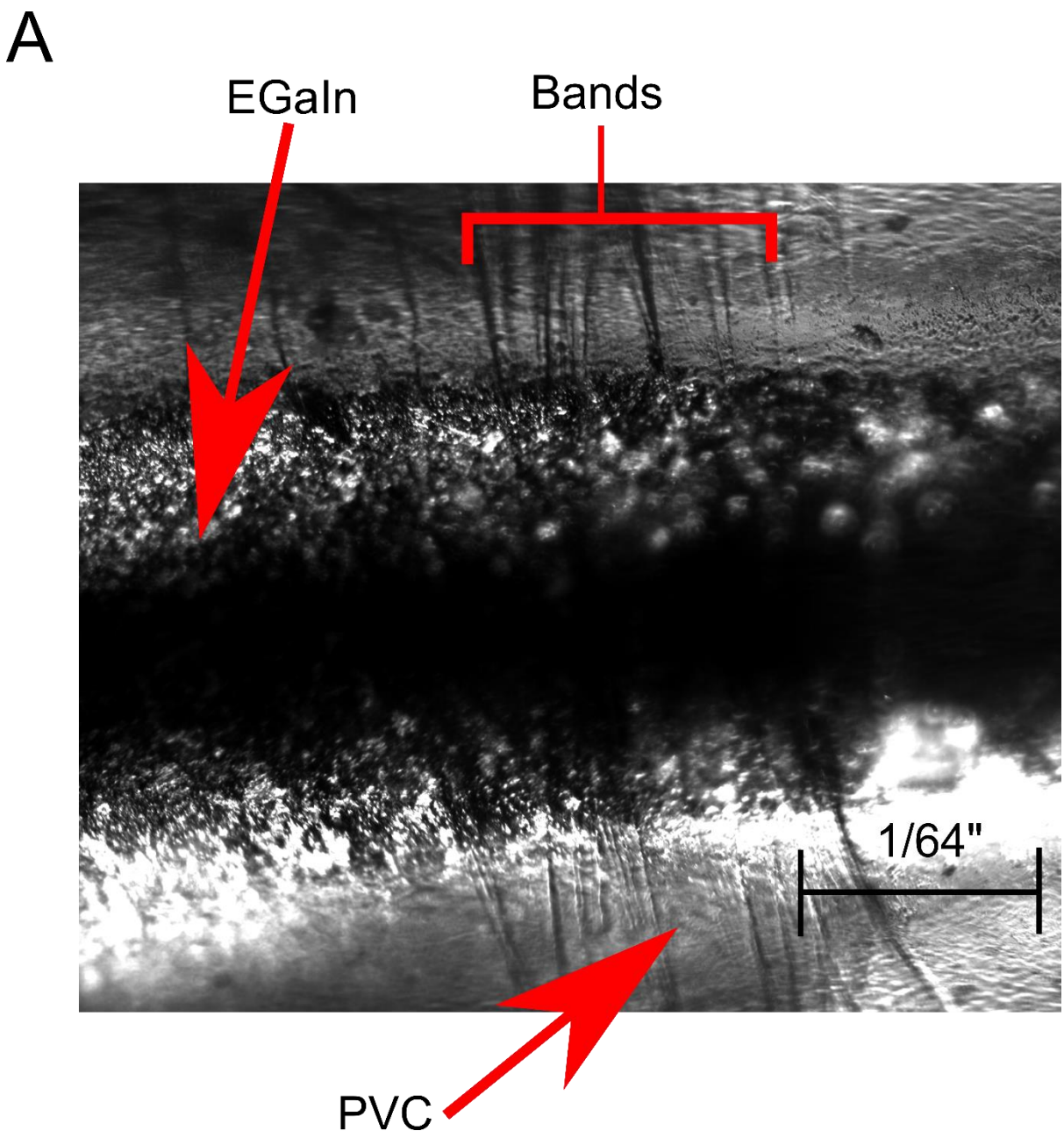

Figure 3: *Microscopy image of PVC tube showing the bands formed at the interface the compressed and uncompressed sensor. On the left is the uncompressed tube, and the on the right is the compressed tube.*

Material Mechanical Characterization

We also found that the materials experienced creep when under a load of above 5 N. We believe that the creep in the material is the cause for hysteresis error when the sensor decompresses. The magnitude of time-dependent creep in a material is temperature dependent. The homologous temperature, defined by equation 8, determines how comparable the creep of the stress value is to the nominal stress value. Creep becomes apparent at a homologous temperature above 0.5 for metals and 0.4 for ceramics. The glass transition temperature is a better indicator of creep for the polymers that we used in our experiments. As a result, some polymers, such as PVC, fluorosilicone, and silicone, experience creep at room temperature.[31]

$$T_H = T/T_{Melting} \tag{8}$$

To show the effect of this phenomenon on our sensors, we compressed our sensor until the system achieved a load of 40 N. We changed this parameter to 55 N for PVC. The material testing machine then held its fixtures position for 2 hours, and we collected the force, impedance, and phase data.

To show how this phenomenon can affect hysteresis, we compressed and decompressed the sensor 48 times in two hours, and then we compressed and decompressed the sensor one time in 2 hours. We hypothesized that the hysteresis would be smaller with a less frequent compression cycle because the material will have more time to follow the material testing compression fixture. The microscopy images of the PVC tube in figure 3 shows the resulting black bands at the interface between the

compression device and the uncompressed tube. These black bands indicate residual stress where the material has not come back to its original state.

Mechanical Testing and Characterization

We tested two sensor setups; a wired and a wireless setup. The wired sensor layout consisted of a resistive tube that connected to a vector network analyzer. We created a base to hold a tube filled with eutectic gallium, and we used a top assembly with two different contact area sizes to apply varying loads to the tube. The fixture that we used for testing had a contact area of 0.8 mm^2. This top assembly applied varying amounts of load to the sensor at a rate of 0.02 mm/s and 1.25 mm/s on average. We performed these tests on three tube types; PVC, fluorosilicone, and silicone. These tests helped us to understand the relationship between hysteresis and rate of loading.

We also tested a fixture with a contact area of 0.2 mm^2 at a rate of 1.25 mm/s on average. This fixture served to help us understand how a decrease in contact area can affect the sensitivity of the sensor. We hypothesized that a smaller load contact area would increase the pressure applied to the sensor. The increase in pressure will decrease the necessary force required to compress the sensor. We tested the same three tubes that we tested with the larger fixture.

For our second setup, we attached the sensor to a pair of inductively coupled coils. The sensor attached to a sensor coil, and the reader coil attached to a vector network analyzer. We tested the sensor with a fixture that had a 0.8 mm^2 contact area. To test the effect of coil distance, we varied the spacing between the sensor coil and reader coil. The material testing machine compressed the same three tubes that it tested in the wired tests. We compressed the silicone rubber and fluorosilicone rubber sensor to a maximum load of 50 N. We compressed the PVC tube sensor to 60 N. Figure 4 shows our two test setups. The response can also change based on the translation of the coil in

the horizontal plane, and the orientation of the coils. However, the scope of this thesis focuses on the normal distance between the coils and keeps the other variables constant.

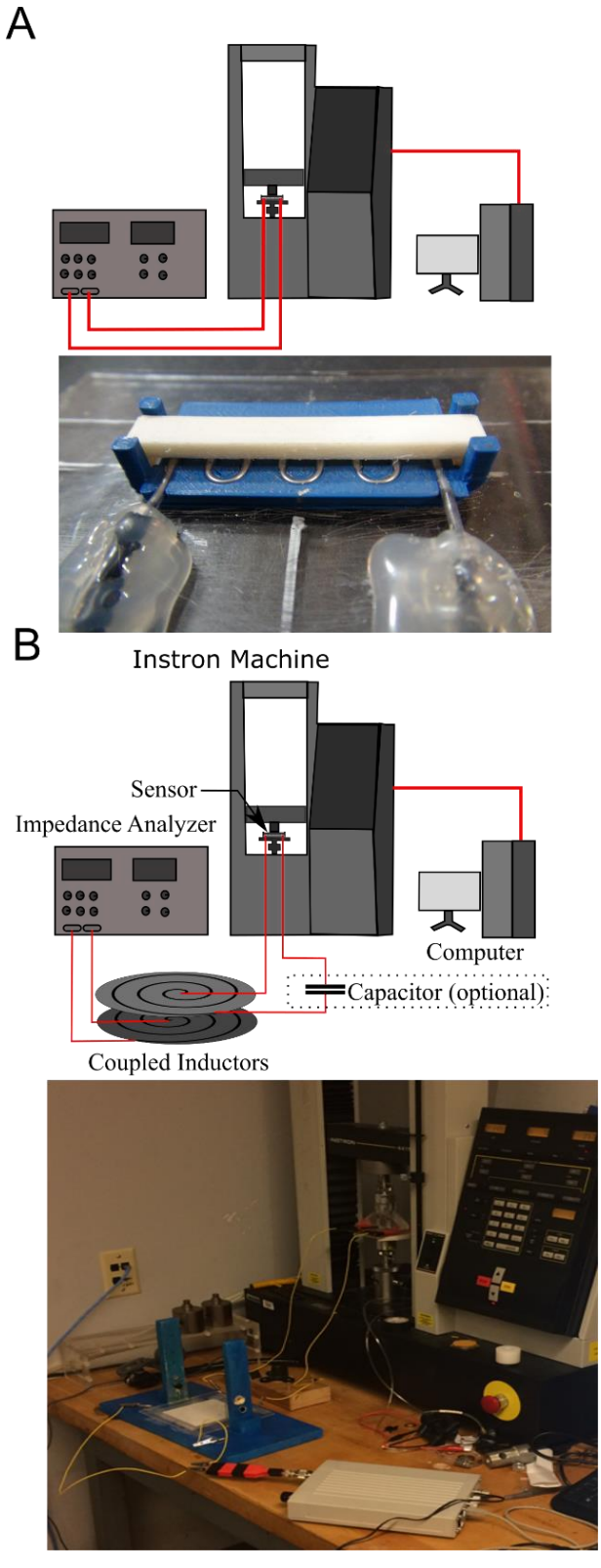

Figure 4: *a) Resistive sensing test Setup b) Wireless sensing test setup (Optional Capacitor).*

Results and Discussions

When we placed the tubes in the environmental chamber, we found that the impedance for each tube stayed constant. The PVC tube had an RMS difference from the mean of 0.011, the fluorosilicone tube had an RMS difference from the mean of 0.008, and the silicone tube had an RMS difference from the mean of 0.014. The PVC tube had a percent drift of 1.3%, the fluorosilicone tube had a percent drift of 0.13%, and the silicone tube had a percent drift of 4.2%. Due to a large volume of noise, we smoothed the reading using a local regression technique in Matlab. These results are not completely conclusive, but they suggest that oxidation of the eGaIn inside the tube may have had a negligible effect on the drift of the impedance readings. Figure 5 shows the results of this test. Based on the results of the oxidation test, we concluded that the hysteresis was not due to the oxidation of the eGaIn.

When we tested for creep in the host materials, we found that this effect was a more apparent cause for the hysteresis. When we compressed the sensor and held the sensor at a constant extension, we found that the load decreased over time, and the impedance increased. Figure 6 shows the results of the creep test. The results show two regions: a region of large change in load and impedance**,** and a region of slow change in load and impedance. In the first region, load quickly and linearly decreases over time, and impedance quickly and linearly increases over time. In the second region, load slowly and linearly decreases over time, and impedance slowly and linearly increases over time. This behavior is consistent with the theory. The theory states that the rate of straining of the material under constant stress will asymptote to a point exponentially.

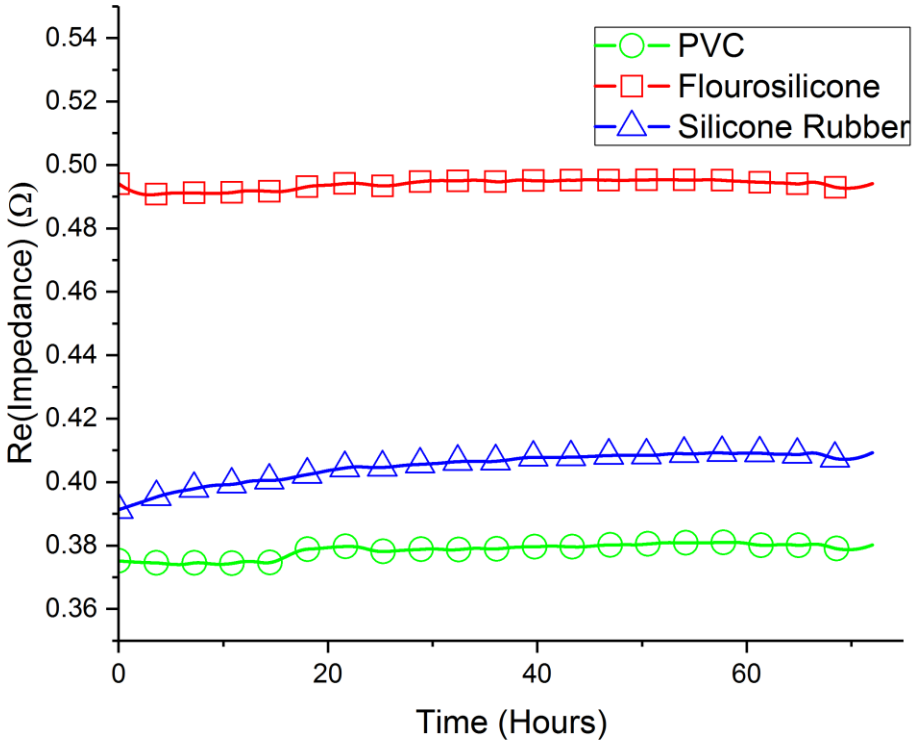

Figure 5: *Impedance of a Silicone Tube, a Fluorosilicone Tube, and a PVC tube over a period of 72 hours shows an insignificant change over time.*

The results in Figure 6 show that the load from the PVC tube, the fluorosilicone tube, and the silicone tube decreased by 41%, 21%, and 11% respectively. The impedance of the PVC tube, the fluorosilicone tube, and the silicone tube increased by 52 %, 82%, and 308% respectively. The interesting point is that the PVC has the largest decrease in load when tested. However, it had the lowest change in impedance. This effect is a result of the difference in elastic modulus. The elastic moduli of PVC is $2.5 \cdot 10^9$ Pa, the elastic moduli of fluorosilicone is $6.9 \cdot 10^7$ Pa, and the elastic moduli of silicone rubber is $7.50 \cdot 10^5$ Pa. As a result, the PVC experienced the smallest change in strain. The PDMS tube experiences the largest change in strain, and the fluorosilicone tube has a change in strain that is in between PVC tube and Silicone rubber tube.

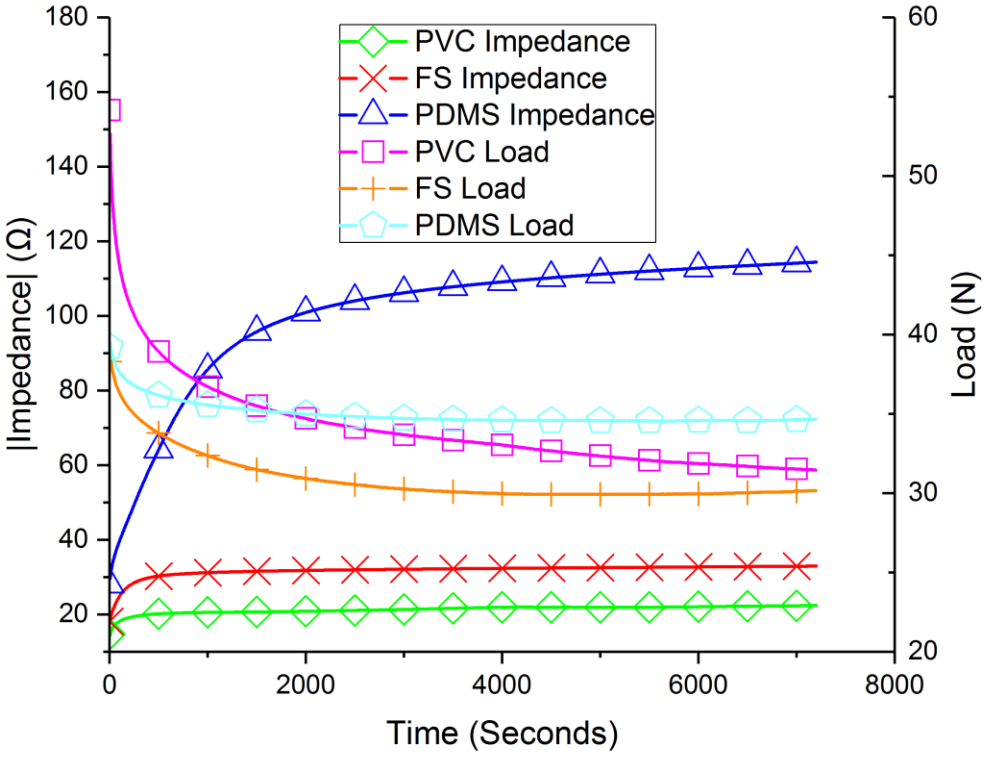

Figure 6: *Constant Load Material Creep Test*

During the second test, shown in Figure 7, we found that a change in the rate of compression from 0.2 mm/min to 0.8 mm/min influenced the behavior of the sensor. When compared to the curves for the compression at 0.8 mm/min, the sensor with a compression rate of 0.2 mm/min experienced a higher maximum impedance. The sensors also experienced a lower maximum force when operated a rate of 0.2 mm/min compared to a rate of 0.8 mm/min. As a result, the slope of the response curve was larger when the rate of compression was 0.2 mm/min when compared to the response of the sensor when compressed at 0.8 mm/min. This result shows that the sensitivity of the sensor is rate dependent.

The rate of compression of the tube also influenced the hysteresis of the response curve. We calculated the hysteresis error by finding the difference between each data point on the compression curve and each corresponding point with the same force value on the decompression curve. The PVC tube had a hysteresis of -0.4448 when compressed at a rate of 1.44 mm/s, and the sensor had a hysteresis of approximately zero when compressed at a rate of 0.031 mm/s. The flourosilicone rubber tube had a hysteresis of 0.8544 when compressed at a rate of 1 mm/s, and the sensor had a hysteresis of 2.8854 when compressed at a rate of 0.022 mm/s. The silicone rubber tube had a hysteresis of -1.6398 when compressed at a rate of 1.16 mm/s, and the sensor had a hysteresis of 1.2797 when compressed at a rate of 0.024 mm/s.

These results confirm that the creep influences the time-dependent nature of the sensors. The sensors are not able to reach as high of a load when compressed at a slower rate because the creep lowers the force as the tube strains. The effect of creep also causes the resistance of the sensor to increase with time. This time-dependent increase in

resistance causes the sensor to have a higher maximum impedance when loaded at a 0.2 mm/min when compared to loading at 0.8 mm/min.

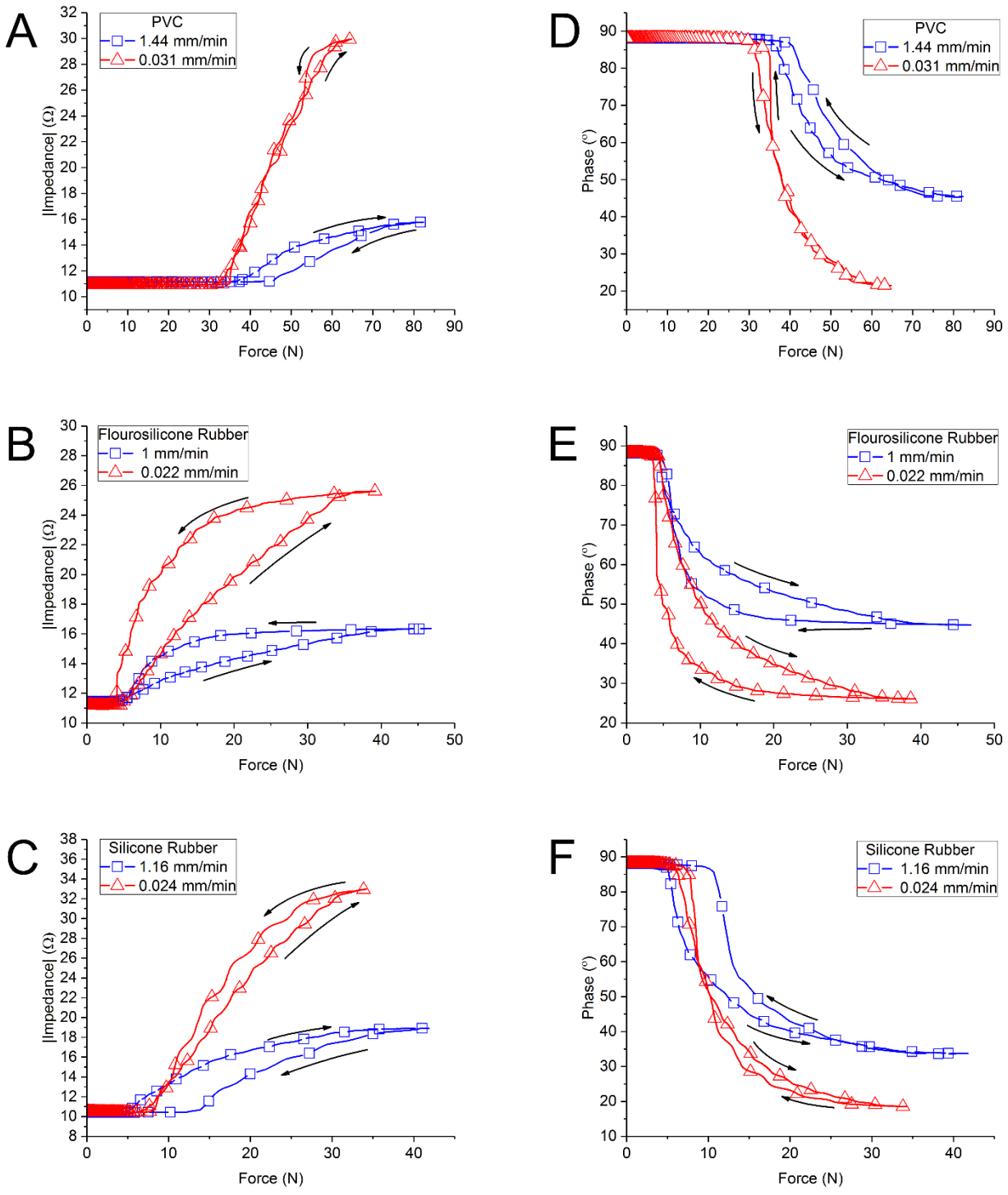

Figure 7: *Effect of loading Speed on impedance response of a soft sensor. a) PVC Sensor b) Flourosilicone Sensor c) Silicone Rubber Sensor.*

In addition to testing for the time-dependent nature of the response curves, we tested the dependence of the curves on the contact area. Figure 8 shows the resulting comparisons between the 0.2 mm^2 and the 0.8 mm^2 fixture arrangements. The results show that for a 0.2 mm^2 contact area, the responses occurred at a lower force than the tests with the 0.8 mm^2 contact area. The response with the PVC occurred at 10 N with a contact area of 0.2 mm^2, and at 40 N with a contact area of 0.8 mm^2. The response with the fluorosilicone occurred at 1 N with a contact area of 0.2 mm^2, and at 5 N with a contact area of 0.8 mm^2. The response with the silicone occurred at 2 N with a contact area of 0.2 mm^2, and at 5 N with a contact area of 0.8 mm^2. These results show that the amount of surface area that is in contact with the tube can influence the sensitivity of the sensors. The decrease in contact area causes the pressure on the tube to increase for each amount of pressure applied to the tube.

In all of these tests, the impedance that is measured is much higher than the resistance of the tubes. The resistance of the tubes was on average 0.4 Ω. However, despite reading the impedance through a direct wired connection, the impedance that we read in the unstrained state is approximately 10 Ω. Also, the phase is approximately 87 degrees in the unstrained state. These readings are a result of the inductance in the resistive tube. This inductance causes the response signal to have an imaginary component that is linearly dependent on the input frequency. This imaginary component causes the impedance to rise with frequency, and the phase to asymptote toward 90° as the frequency increases. Figure 9 shows these effects.

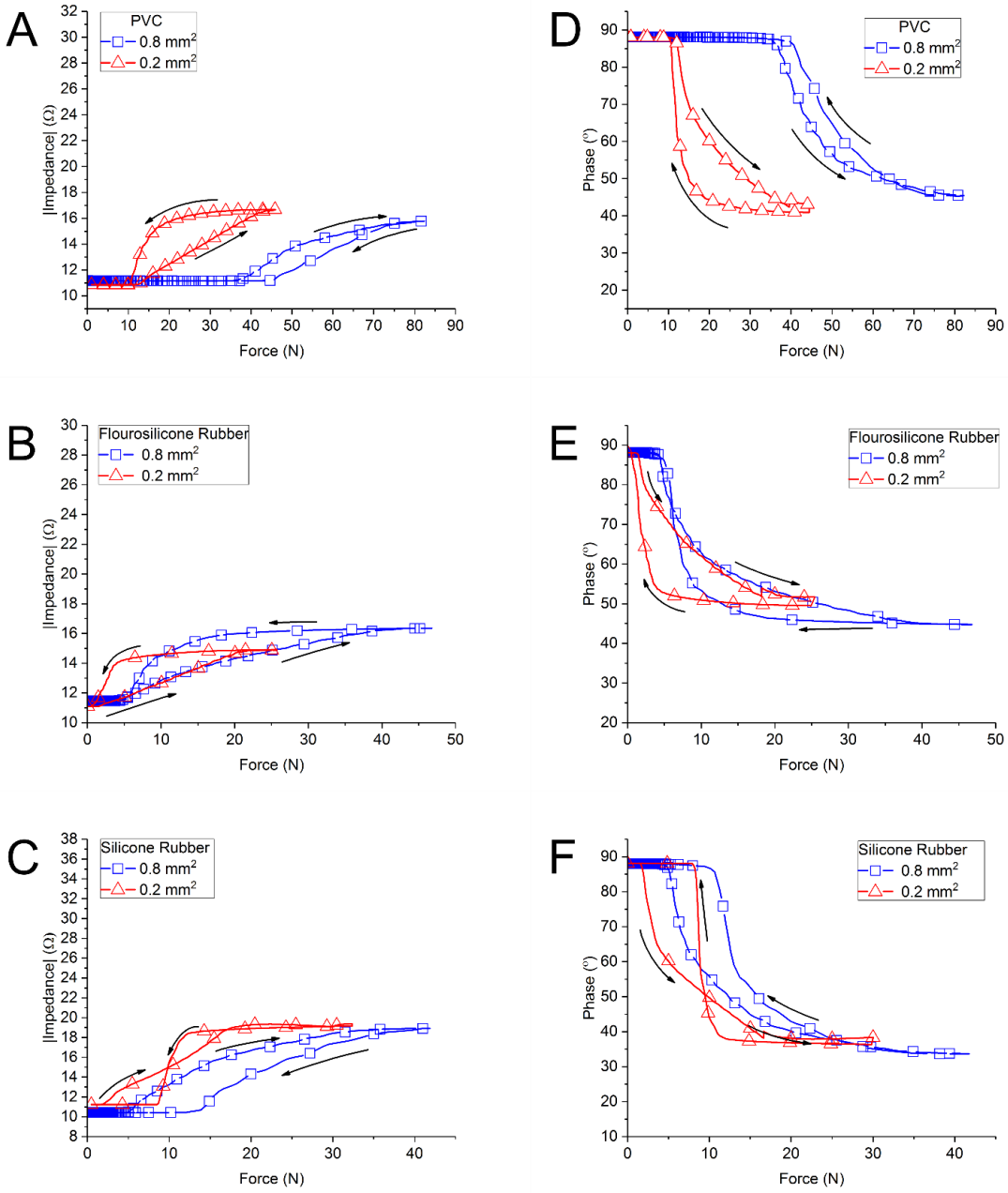

Figure 8: *Comparison of the wired responses between a 0.8mm^2 and a 0.2 mm^2 contact area. a) PVC sensor impedance response b) Fluorosilicone rubber sensor impedance response c) Silicone rubber sensor impedance response. d) PVC sensor phase response e) Fluorosilicone rubber sensor phase response f) Silicone rubber sensor phase response.*

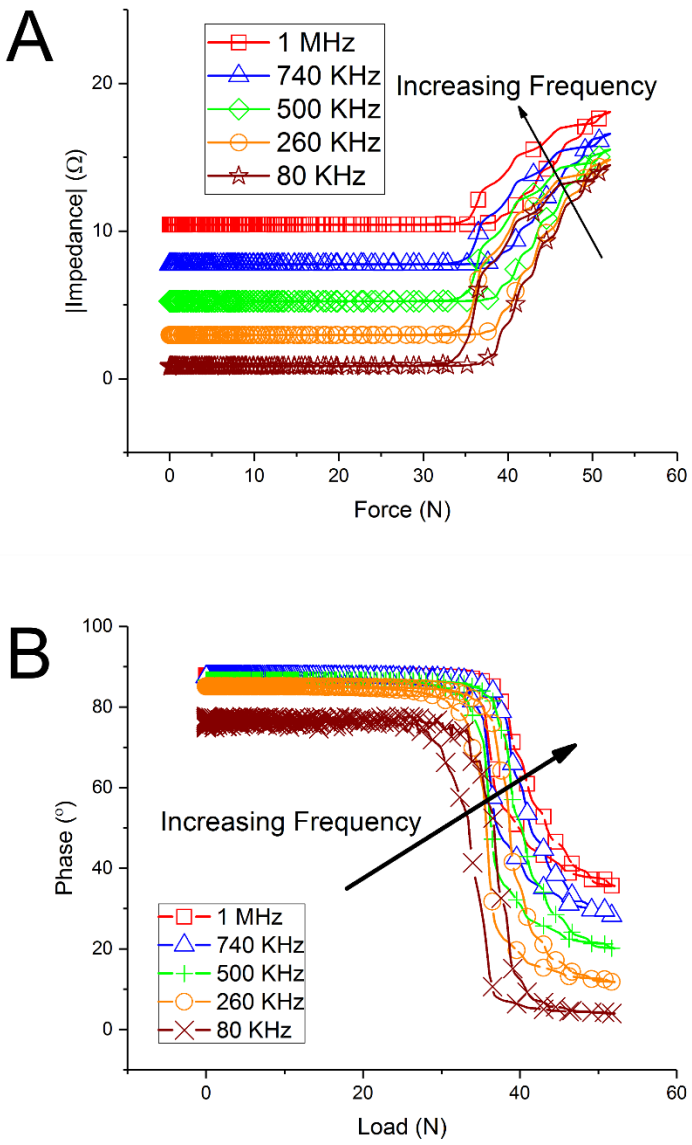

Figure 9: *Effect of stimulus frequency on phase response of a PVC soft sensor.*

The wireless results show how the amount of separation between the sensor circuit and the reader circuit affects the sensitivity of the sensor. Figure 10 shows the results of the experiment in which we tested the wireless setup with two coil separation distances. The results show that the sensitivity of the PVC, fluorosilicone, and PDMS sensors were 69.2 mΩ/N, 19.7 mΩ/N, and 53.5 mΩ/N respectively when we separated the coils by ¼". When the coils were separated by 1", the sensitivity of the PVC, fluorosilicone, and PDMS sensors were 4.5 mΩ/N, 4.6 mΩ/N, and 7.8 mΩ/N respectively. For non-resonant wireless sensing, the phase experiences a higher sensitivity than the impedance. When the coils were ¼" apart, the sensitivity of the PVC, fluorosilicone, and PDMS sensors were 270 millidegrees /N, 78 millidegrees /N, and 136 millidegrees /N respectively. When the coils were 1" apart, the sensitivity of the PVC, fluorosilicone, and PDMS sensors were 9.6 millidegrees /N, 10 millidegrees /N, and 16 millidegrees /N respectively.

These results show that we were able to detect pressure based upon a change in impedance of the system. The phase experiences a higher sensitivity because the impedance due to the inductance of the sensor is much larger than the impedance due to the resistance of the sensor. As a result, the phase starts at 90 degrees and decreases while the resistance in the sensor increases due to an applied load.

The sensitivity of the sensor due to an applied force also decreases due to the increase in separation distance. This decrease in sensitivity is a result of the change in mutual inductance. This mutual inductance changes because the flux of magnetic field traveling through the coil. These results indicate that a fixed distance between coils is ideal when detecting pressure in non-resonant sensors.

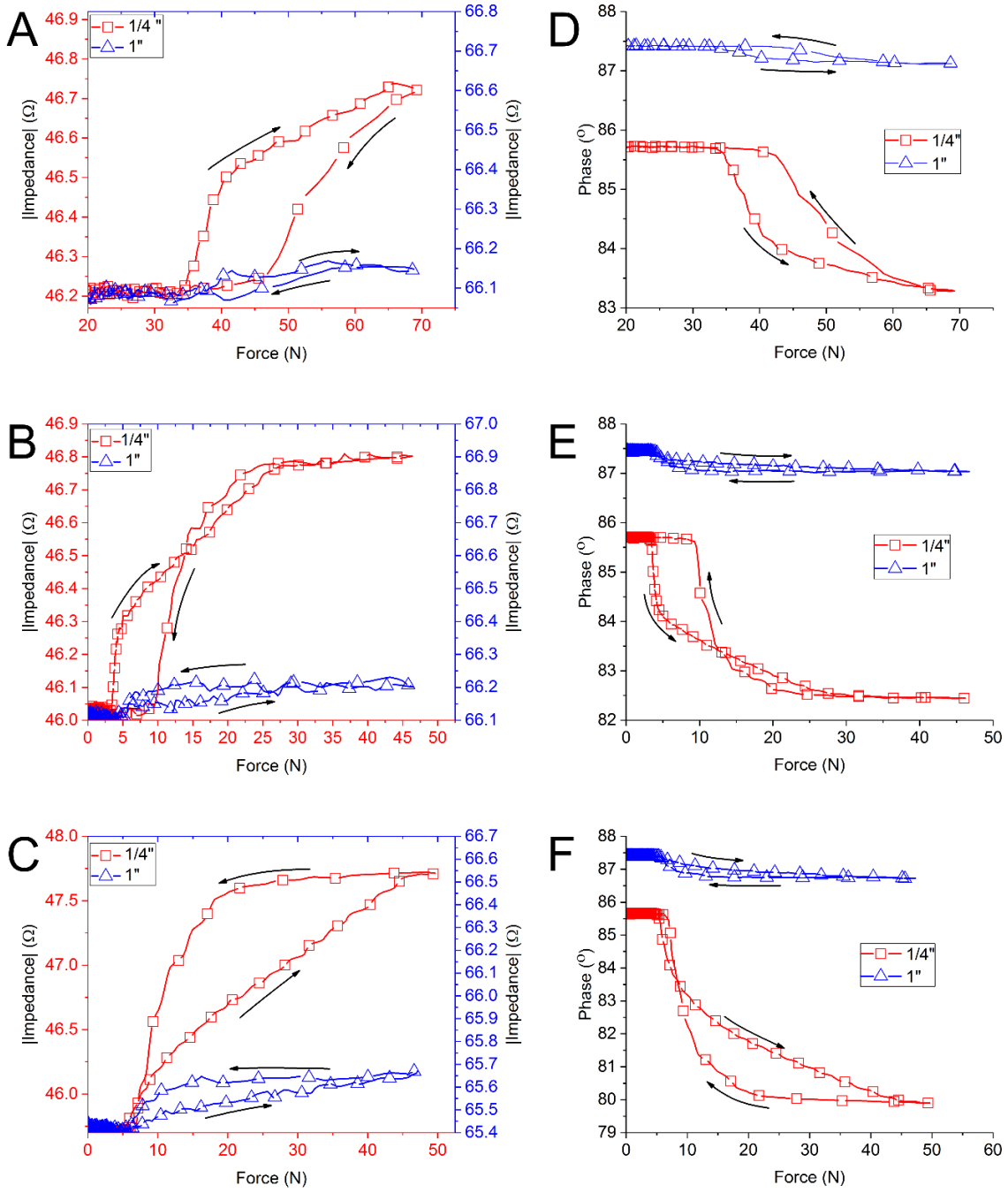

Figure 10: *Wirelessly Inductive coupled sensor with inductors placed ¼" and 1" apart a) PVC sensor impedance response b) Fluorosilicone rubber sensor impedance response c) Silicone rubber sensor impedance response. d) PVC sensor phase response e) Fluorosilicone rubber sensor phase response f) Silicone rubber sensor phase response.*

When we compare the sensitivity of our sensor, to the sensitivity other sensors if the field, we find that our wired sensors have a comparable sensitivity to the sensors of other papers, However, our wireless sensor has a much lower sensitivity when compared to other sensors in the field. The low sensitivity of our wireless soft sensor is a result of the loss of energy as the signal transmits through the air. Table 1 compares the performance of our sensor to other papers. In the table, we compared the sensitivity of the wired soft sensors of Young Lee Park, [2] Robert Wood,[3] and our wireless sensor.

We approximated sensitivity by dividing the change in resistance by the change in pressure. Since we measured the resistance of our sensors as a function of force, we needed to divide the force readings by the contact area between the tube and the fixture. Equation 9 gives the sensitivity of the sensor.

$$Sensitivity = \frac{R_{Max} - R_{Min}}{P_{Max} - P_{Min}} \tag{9}$$

Where R_{Max} is the maximum resistance, R_{Min} is the initial resistance, P_{Max} is the maximum pressure, and P_{Min} is the pressure at which the sensor starts to become sensitive. To maintain consistency between experiments, we calculated the real component of the impedance for both the wired and the wireless sensor setups. We used the real component of impedance and the applied pressure throughout all of the sensitivity calculations. We calculated the real component of the impedance by multiplying the magnitude of the impedance by the cosine of the phase angle. Equation 10 shows the calculation for the real component of the impedance. Where |Z| is the magnitude of the impedance, and φ is the phase angle.

$$Re(|Z|) = |Z| \cdot \cos(\varphi) \quad (10)$$

The results show that the sensitivity of our wired sensors is comparable to the sensitivities of the other papers. Although the sensitivity of our wireless sensor is lower than the sensors of other papers, our sensors can read pressure and can operate through acrylic via a wireless inductive coupling. Table 1 shows the sensitivity of our sensors made from PDMS as well as the sensors from a paper from Park et al. and a paper by Kramer et al.

Paper	**Park Wired Sensor**	**Kramer Wired Sensor**	**Our Wired Sensor**	**Our Wireless Sensor**
Range (Ω)	6.000	0.060	30.876	4.899
Sensitivity (Ω/Kpa)	0.400	0.006	0.021	0.002

Table 1: *Comparison Matrix of sensor performance from previous works, all sensors compared in this table have PDMS as the encapsulating matrix.*

A smart gasket is ideal for non-resonant soft sensor detection with a fixed coil separation distance. We created a gasket by embedding our sensor in Ecoflex. Figure 11 shows the gasket design and the results from the compression of the gasket. We made the sensor by injecting eGaIn into a tube made from PDMS using the method that we used to create the previously tested sensors. We then soldered the leads at the ends of the sensor tube to a coil. To embed the sensor inside the gasket, we built a mold around the cap of the device that we wanted to seal, and we placed the sensor inside the mold. We poured Ecoflex into the mold, and we let it cure for 3 hours at room temperature. We then placed a cap on the device we wanted to seal, and we placed the reader on top of the cap. When we screwed the cap onto the assembly, it compressed the gasket, and the phase response dropped.

Another application of resistive wireless soft sensing is a push button sensor. Figure 12 shows this application. The button consists of a microfluidic tube encased in a holder that guides the button as it moves upward or downward. The button has two pins that contact the tube with a surface area of 0.2 mm^2. As we showed earlier, a smaller contact area makes the button sensitive to lower forces. When the button compresses the tube, the magnitude of the impedance rises, and the phase drops. Figure 12b shows the change in the impedance and phase as a function of frequency. The reader can still measure this effect with an acrylic sheet between the coils as well. The ability to measure impedance through a non-conductive material demonstrates that it is possible to embed the button into consumer devices such as keyboards, computers, and cell phones.

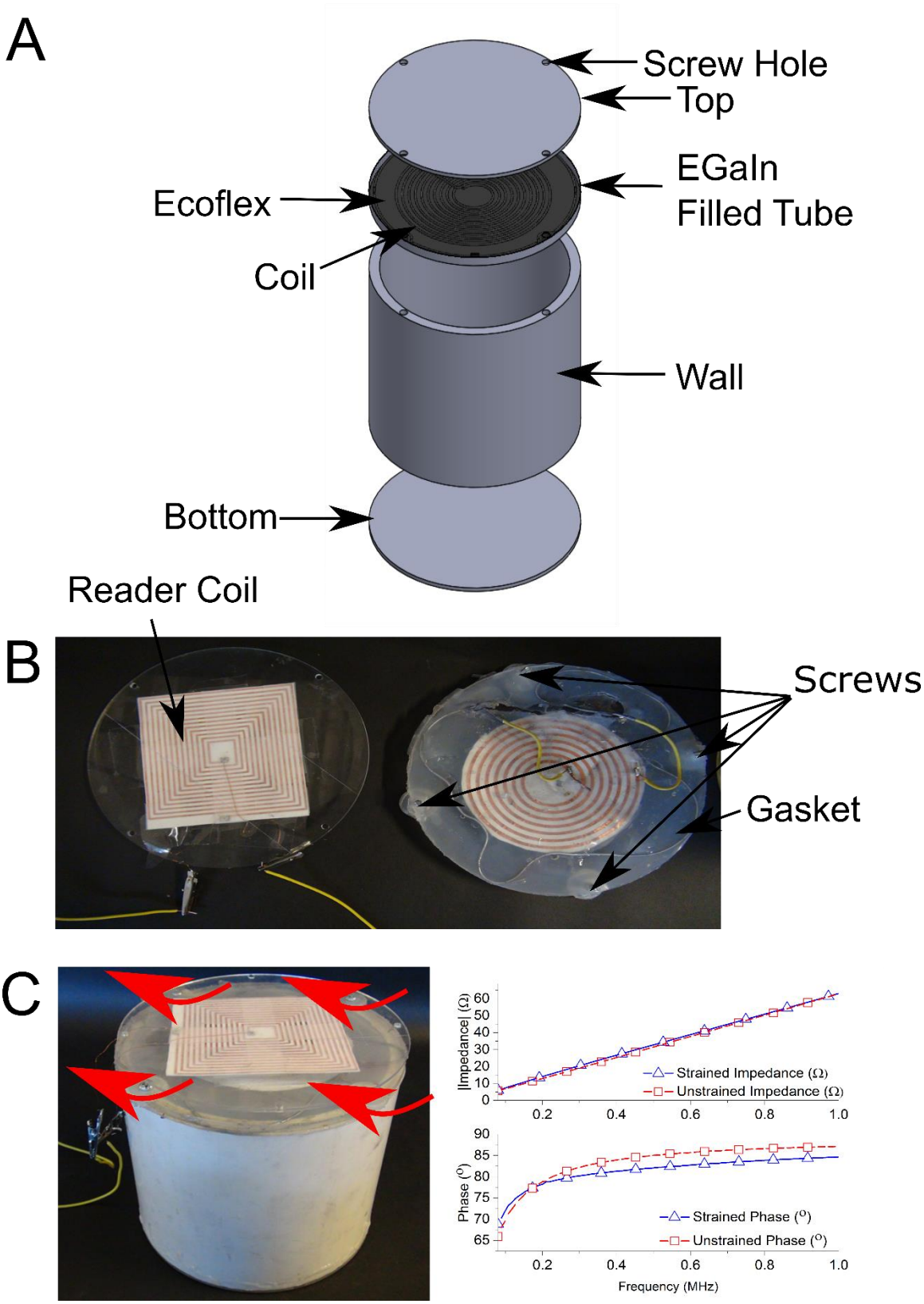

Figure 11: *a) Gasket and tub assembly b) Gasket and reader coil c) Compressed gasket and graph comparing uncompressed and compressed gasket response.*

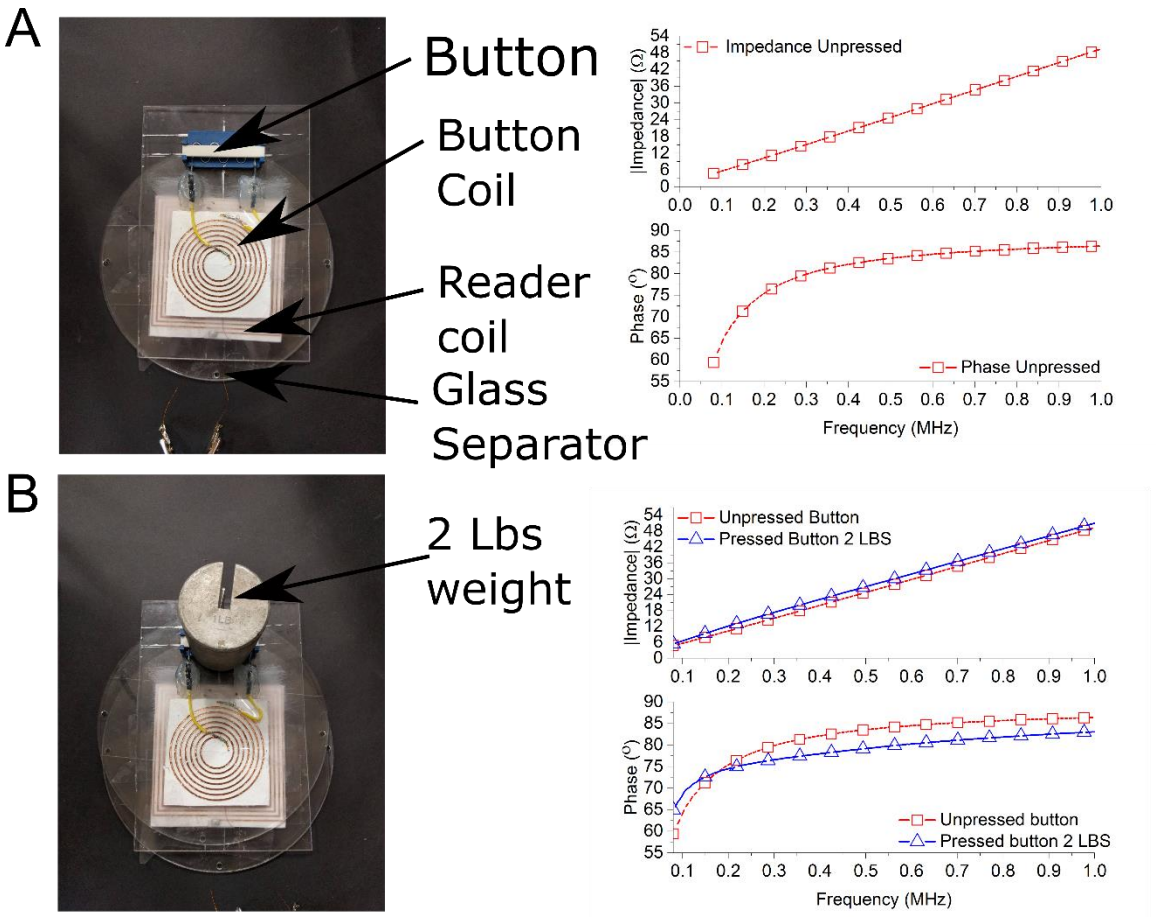

Figure 12: Wireless button impedance before and after compression *a) Uncompressed button b) Compressed button*

Conclusions and Future Work

In this work, we discuss the effects of resistive wireless strain sensing from two angles. We discuss the effects on a resistive tube that is undergoing a mechanical loading, and the application of these effects in an inductively coupled setup. We were able to prove the applicability of soft pressure sensors to wireless sensing. While investigating the effects of loading on soft sensors, we discovered that the creep in the host material caused the hysteresis of the setup to be rate dependent. We perform load tests at an average rate of 1.25 mm/min and then 0.025 mm/min. We found that the sensitivity of the response increased with a decreased rate of loading. While performing wireless tests, we found that we were able to detect a sensitivity of up to 270 millidegrees/N.

Future work on this subject will be to characterize the relationship of the orientation of the coils concerning each other and the sensitivity, as well as investigating ways to detect multiple loadings at once. The ability to detect multiple sensors will allow the development of haptic devices such as smart keyboards with multiple keys, and smart gaskets that can detect the precise locations of failure. We can conclude that soft inductively coupled wireless sensors will be of future use to the fields of soft robotics, smart seals, and haptics. This technology will be a benefit in the future, and development and progress are promising.

Appendix

Drift test interpolation code

```
clear;
clc;
PVCImpedanceDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\ImpedancePVCFC 55.txt',',');
PVCphaseDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\PhasePVCFC 55.txt',',');
PVCLoadDriftstraincontrol=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat edge\PVC
FC Load 55.xlsx');

FSImpedanceDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\ImpedanceFSFC 27-Jul-2017 11-22-18.txt',',');
FSphaseDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\PhaseFSFC 27-Jul-2017 11-22-20.txt',',');
FSLoadDriftstraincontrol=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat edge\FS FC
Load.xlsx');

PDMSImpedanceDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\ImpedancePDMSFC 27-Jul-2017 13-38-57.txt',',');
PDMSphaseDriftstraincontrol = importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat
edge\PhasePDMSFC 27-Jul-2017 13-38-58.txt',',');
PDMSLoadDriftstraincontrol=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\flat edge\PDMS
FC Load.xlsx');
%Data cleanup
[PVCIMPPKS,PVCIMPLOCS]=findpeaks(PVCImpedanceDriftstraincontrol(51,2:end),'
MinPeakHeight',25);
for i=1:length(PVCIMPLOCS)

PVCImpedanceDriftstraincontrol(:,PVCIMPLOCS(i)+1)=[PVCImpedanceDriftstraincont
rol(1,(PVCIMPLOCS(i)+1));PVCImpedanceDriftstraincontrol(2:end,(PVCIMPLOCS(i)-
2))];
end

PVCLoadDriftstraincontrol=PVCLoadDriftstraincontrol(3134:end,:);
```

```
FSLoadDriftstraincontrol=FSLoadDriftstraincontrol(3293:end,:);
PDMSLoadDriftstraincontrol=PDMSLoadDriftstraincontrol(2568:end,:);
figure(1);
plot(PVCLoadDriftstraincontrol(:,1),abs(PVCLoadDriftstraincontrol(:,3)))
hold on;
plot(FSLoadDriftstraincontrol(:,1),abs(FSLoadDriftstraincontrol(:,3)))
plot(PDMSLoadDriftstraincontrol(:,1),abs(PDMSLoadDriftstraincontrol(:,3)))
hold off

PVCI=find(PVCImpedanceDriftstraincontrol(1,:)<PVCLoadDriftstraincontrol(1,1));
PVCImpedanceDriftstraincontrol=PVCImpedanceDriftstraincontrol(:,PVCI(end):end);
FSI=find(FSImpedanceDriftstraincontrol(1,:)<FSLoadDriftstraincontrol(1,1));
FSImpedanceDriftstraincontrol=FSImpedanceDriftstraincontrol(:,FSI(end):end);
PDMSI=find(PDMSImpedanceDriftstraincontrol(1,:)<PDMSLoadDriftstraincontrol(1,1))
;
PDMSImpedanceDriftstraincontrol=PDMSImpedanceDriftstraincontrol(:,PDMSI(end):e
nd);

PVCIMP=[(PVCImpedanceDriftstraincontrol(1,:)-
PVCImpedanceDriftstraincontrol(1,1));PVCImpedanceDriftstraincontrol(51,:)];
PVCLD=[(PVCLoadDriftstraincontrol(:,1)-
PVCLoadDriftstraincontrol(1,1)),abs(PVCLoadDriftstraincontrol(:,3))];

FSIMP=[(FSImpedanceDriftstraincontrol(1,:)-
FSImpedanceDriftstraincontrol(1,1));FSImpedanceDriftstraincontrol(51,:)];
FSLD=[(FSLoadDriftstraincontrol(:,1)-
FSLoadDriftstraincontrol(1,1)),abs(FSLoadDriftstraincontrol(:,3))];

PDMSIMP=[(PDMSImpedanceDriftstraincontrol(1,:)-
PDMSImpedanceDriftstraincontrol(1,1));PDMSImpedanceDriftstraincontrol(51,:)];
PDMSLD=[(PDMSLoadDriftstraincontrol(:,1)-
PDMSLoadDriftstraincontrol(1,1)),abs(PDMSLoadDriftstraincontrol(:,3))];

reftime=0:5:7200;
PVCIMPfinal=interp1(PVCIMP(1,:),PVCIMP(2,:),reftime);
PVCLDfinal=interp1(PVCLD(:,1),PVCLD(:,2),reftime);
plot(reftime,PVCLDfinal)

FSIMPfinal=interp1(FSIMP(1,:),FSIMP(2,:),reftime);
FSLDfinal=interp1(FSLD(:,1),FSLD(:,2),reftime);
plot(reftime,FSLDfinal)

PDMSIMPfinal=interp1(PDMSIMP(1,:),PDMSIMP(2,:),reftime);
PDMSLDfinal=interp1(PDMSLD(:,1),PDMSLD(:,2),reftime);
plot(reftime,PDMSLDfinal)
```

```
IMPLD=[reftime;PVCIMPfinal;FSIMPfinal;PDMSIMPfinal;PVCLDfinal;FSLDfinal;PD
MSLDfinal];
a=1440;
PVCLoadPC=100.*abs((IMPLD(5,1)-IMPLD(5,a)))/IMPLD(5,1);
PVCIMPPC=100.*abs((IMPLD(2,1)-IMPLD(2,a)))/IMPLD(2,1);
FSLoadPC=100.*abs((IMPLD(6,1)-IMPLD(6,a)))/IMPLD(6,1);
FSIMPPC=100.*abs((IMPLD(3,1)-IMPLD(3,a)))/IMPLD(3,1);
PDMSLoadPC=100.*abs((IMPLD(7,1)-IMPLD(7,a)))/IMPLD(7,1);
PDMSIMPPC=100.*abs((IMPLD(4,1)-IMPLD(4,a)))/IMPLD(4,1);
disp(['PVCLoadPC=' num2str(PVCLoadPC)])
disp(['PVCIMPPC=' num2str(PVCIMPPC)])
disp(['FSLoadPC=' num2str(FSLoadPC)])
disp(['FSIMPPC=' num2str(FSIMPPC)])
disp(['PDMSLoadPC=' num2str(PDMSLoadPC)])
disp(['PDMSIMPPC=' num2str(PDMSIMPPC)])

figure(3);
yyaxis left
plot(IMPLD(1,:),IMPLD(2,:),IMPLD(1,:),IMPLD(3,:),IMPLD(1,:),IMPLD(4,:))
yyaxis right
plot(IMPLD(1,:),IMPLD(5,:),IMPLD(1,:),IMPLD(6,:),IMPLD(1,:),IMPLD(7,:))
THIMPLD=(54-35).*(exp(-.006.*IMPLD(1,:)))+35;
figure(4);
plot(IMPLD(1,:),IMPLD(5,:),'b',IMPLD(1,:),THIMPLD,'r--')
```

Rate test interpolation code

```
%% PVC
clear;
clc;
%One Cycle
% Impedance
PVCIMPonecycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\ImpedancePVCSCSEonecycleredo 09-Aug-2017 13-14-08.txt',',');
PVCIMPtimeonecycle=PVCIMPonecycle(1,2:end);
PVCIMP1MHzonecycle=PVCIMPonecycle(51,2:end);
%Phase
PVCPhaseonecycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\PhasePVCSCSEonecycleredo 09-Aug-2017 13-14-09.txt',',');
PVCPhasetimeonecycle=PVCPhaseonecycle(1,2:end);
PVCPhase1MHzonecycle=PVCPhaseonecycle(51,2:end);
%Load
PVCLoadtimeonecycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one cycle\PVC
Load one cycle redo.xlsx','PVC 1.85 mm max SC 1  cycles re','A2:A144149');
```

```
PVCExtensiononecycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one cycle\PVC
Load one cycle redo.xlsx','PVC 1.85 mm max SC 1  cycles re','B2:B144149');
PVCLoadonecycle=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\PVC Load one cycle redo.xlsx','PVC 1.85 mm
max SC 1  cycles re','C2:C144149');

PVCLoadonecycle=abs(PVCLoadonecycle);
PVCLoadMatrixonecycle=[PVCLoadtimeonecycle,PVCExtensiononecycle,abs(PVCExte
nsiononecycle),PVCLoadonecycle,abs(PVCLoadonecycle)];

PVCIMP1MHzMatrixonecycle=[PVCIMP1MHzonecycle;PVCPhase1MHzonecycle;PV
CIMPtimeonecycle];
PVCIMP1MHzMatrixonecycle=PVCIMP1MHzMatrixonecycle';
PVCPhase1MHzMatrixonecycle=[abs(PVCPhase1MHzonecycle-
PVCPhase1MHzonecycle(1));PVCIMP1MHzonecycle;PVCIMPtimeonecycle];
PVCPhase1MHzMatrixonecycle=PVCPhase1MHzMatrixonecycle';
%cleaning up data
[a,b]=findpeaks(PVCIMP1MHzMatrixonecycle(:,1),'MinPeakHeight',20);
for i=1:length(b)
   PVCIMP1MHzMatrixonecycle(b(i),1)=PVCIMP1MHzMatrixonecycle((b(i)-1),1);
end

[PVCIMP1MHzinteronecycle]=analdata4(PVCIMP1MHzMatrixonecycle,PVCLoadMatr
ixonecycle,27,64.7);
[PVCPhase1MHzinteronecycle]=analdata4(PVCPhase1MHzMatrixonecycle,PVCLoadM
atrixonecycle,(66.5),64.7);
PVCPhase1MHzinteronecycle(:,1:2:end)=-
PVCPhase1MHzinteronecycle(:,1:2:end)+PVCPhase1MHzonecycle(1);
PVConecycleavghysterisis=avghyst(PVCIMP1MHzinteronecycle,50)
% 48 Cycle
% Impedance
PVCIMPfortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\ImpedancePVCSCSEfortyeightcycleredo 09-Aug-2017 16-56-59.txt',',');
PVCIMPtimefortyeightcycle=PVCIMPfortyeightcycle(1,2:end);
PVCIMP1MHzfortyeightcycle=PVCIMPfortyeightcycle(51,2:end);
%Phase
PVCPhasefortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PhasePVCSCSEfortyeightcycleredo 09-Aug-2017 16-57-00.txt',',');
PVCPhasetimefortyeightcycle=PVCPhasefortyeightcycle(1,2:end);
PVCPhase1MHzfortyeightcycle=PVCPhasefortyeightcycle(51,2:end);
%Load
```

```
PVCLoadtimefortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\PVC
Load 48 cycles 1.85 redo.xlsx','PVC 1.85 mm max SC 48 cycles re','A2:A144678');
PVCExtensionfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\PVC
Load 48 cycles 1.85 redo.xlsx','PVC 1.85 mm max SC 48 cycles re','B2:B144678');
PVCLoadfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\PVC
Load 48 cycles 1.85 redo.xlsx','PVC 1.85 mm max SC 48 cycles re','C2:C144678');

PVCLoadfortyeightcycle=abs(PVCLoadfortyeightcycle);
PVCLoadMatrixfortyeightcycle=[PVCLoadtimefortyeightcycle,PVCExtensionfortyeight
cycle,abs(PVCExtensionfortyeightcycle),PVCLoadfortyeightcycle,abs(PVCLoadfortyeig
htcycle)];

PVCIMP1MHzMatrixfortyeightcycle=[PVCIMP1MHzfortyeightcycle;PVCPhase1MHzf
ortyeightcycle;PVCIMPtimefortyeightcycle];
PVCIMP1MHzMatrixfortyeightcycle=PVCIMP1MHzMatrixfortyeightcycle';
PVCPhase1MHzMatrixfortyeightcycle=[abs(PVCPhase1MHzfortyeightcycle-
PVCPhase1MHzfortyeightcycle(1));PVCIMP1MHzfortyeightcycle;PVCIMPtimefortyeig
htcycle];
PVCPhase1MHzMatrixfortyeightcycle=PVCPhase1MHzMatrixfortyeightcycle';
%cleaning up data
[a,b]=findpeaks(PVCIMP1MHzMatrixfortyeightcycle(:,1),'MinPeakHeight',20);
for i=1:length(b)

PVCIMP1MHzMatrixfortyeightcycle(b(i),1)=PVCIMP1MHzMatrixfortyeightcycle((b(i)-
1),1);
end

[PVCIMP1MHzinterfortyeightcycle]=analdata5(PVCIMP1MHzMatrixfortyeightcycle,P
VCLoadMatrixfortyeightcycle,15,75);
[PVCPhase1MHzinterfortyeightcycle]=analdata5(PVCPhase1MHzMatrixfortyeightcycle,
PVCLoadMatrixfortyeightcycle,(40),75);
PVCPhase1MHzinterfortyeightcycle(:,2)=-PVCPhase1MHzinterfortyeightcycle(:,2);
PVCPhase1MHzinterfortyeightcycle(:,2)=PVCPhase1MHzinterfortyeightcycle(:,2)+PVC
Phase1MHzfortyeightcycle(1,1);

PVCfortyeightcycleavghysterisis=avghyst([PVCIMP1MHzinterfortyeightcycle(round(2.*
length(PVCIMP1MHzinterfortyeightcycle)/48):round(3.*length(PVCIMP1MHzinterfort
yeightcycle)/48),2),PVCIMP1MHzinterfortyeightcycle(round(2.*length(PVCIMP1MHzi
nterfortyeightcycle)/48):round(3.*length(PVCIMP1MHzinterfortyeightcycle)/48),1)],75)
PVCerrorreadymatrix=errorreadymatrix(PVCIMP1MHzinterfortyeightcycle,50);
PVCerrorreadymatrixfinalc=mean(PVCerrorreadymatrix(2:2:end,:));
PVCerrorreadymatrixfinald=mean(PVCerrorreadymatrix(3:2:end,:));
a=350;
```

```
b=400;
PVCerrc=std(PVCerrorreadymatrix(2:2:end,1:a:end));
PVCerrd=std(PVCerrorreadymatrix(3:2:end,1:b:end));
figure(1);
plot(PVCerrorreadymatrix(1,:),PVCerrorreadymatrixfinalc,PVCerrorreadymatrix(1,:),PV
Cerrorreadymatrixfinald)
figure(2);
errorbar(PVCerrorreadymatrix(1,1:a:end),PVCerrorreadymatrixfinalc(1:a:end),1.671.*P
VCerrc)
hold on;
errorbar(PVCerrorreadymatrix(1,1:b:end),PVCerrorreadymatrixfinald(1:b:end),1.671.*P
VCerrd)
hold off;

%% FS
%One Cycle
% Impedance
FSIMPonecycle=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\ImpedanceFSSCSEonecycle 03-Aug-2017 14-
53-58.txt',',');
FSIMPtimeonecycle=FSIMPonecycle(1,2:end);
FSIMP1MHzonecycle=FSIMPonecycle(51,2:end);
%Phase
FSPhaseonecycle=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\PhaseFSSCSEonecycle 03-Aug-2017 14-54-
00.txt',',');
FSPhasetimeonecycle=FSPhaseonecycle(1,2:end);
FSPhase1MHzonecycle=FSPhaseonecycle(51,2:end);
%Load
FSLoadtimeonecycle=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\FS Load one cycle.xlsx','FS 1.3 N max SC 1
cycle.is_ccyc','A2:A143703');
FSExtensiononecycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one cycle\FS
Load one cycle.xlsx','FS 1.3 N max SC 1 cycle.is_ccyc','B2:B143703');
FSLoadonecycle=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\FS Load one cycle.xlsx','FS 1.3 N max SC 1
cycle.is_ccyc','C2:C143703');

FSLoadonecycle=abs(FSLoadonecycle);
FSLoadMatrixonecycle=[FSLoadtimeonecycle,FSExtensiononecycle,abs(FSExtensionon
ecycle),FSLoadonecycle,abs(FSLoadonecycle)];

FSIMP1MHzMatrixonecycle=[FSIMP1MHzonecycle;FSPhase1MHzonecycle;FSIMPtim
eonecycle];
FSIMP1MHzMatrixonecycle=FSIMP1MHzMatrixonecycle';
```

```
FSPhase1MHzMatrixonecycle=[abs(FSPhase1MHzonecycle-
FSPhase1MHzonecycle(1));FSIMP1MHzonecycle;FSIMPtimeonecycle];
FSPhase1MHzMatrixonecycle=FSPhase1MHzMatrixonecycle';


[FSIMP1MHzinteronecycle]=analdata4(FSIMP1MHzMatrixonecycle,FSLoadMatrixonec
ycle,25.6,39);
[FSPhase1MHzinteronecycle]=analdata4(FSPhase1MHzMatrixonecycle,FSLoadMatrixo
necycle,(62.06),39);
FSPhase1MHzinteronecycle(:,1)=-
FSPhase1MHzinteronecycle(:,1)+FSPhase1MHzonecycle(1);
FSonecycleavghysterisis=avghyst(FSIMP1MHzinteronecycle,35)

% 48 Cycle
% Impedance
FSIMPfortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\ImpedanceFSSCSEfortyeightcycle 03-Aug-2017 17-07-21.txt',',');
FSIMPtimefortyeightcycle=FSIMPfortyeightcycle(1,2:end);
FSIMP1MHzfortyeightcycle=FSIMPfortyeightcycle(51,2:end);
%Phase
FSPhasefortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PhaseFSSCSEfortyeightcycle 03-Aug-2017 17-07-22.txt',',');
FSPhasetimefortyeightcycle=FSPhasefortyeightcycle(1,2:end);
FSPhase1MHzfortyeightcycle=FSPhasefortyeightcycle(51,2:end);
%Load
FSLoadtimefortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\FS
Load 48 cycles 1.3.xlsx','FS 1.3 N max SC 48 cycles.is_cc','A2:A145143');
FSExtensionfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\FS
Load 48 cycles 1.3.xlsx','FS 1.3 N max SC 48 cycles.is_cc','B2:B145143');
FSLoadfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48 cycles\FS
Load 48 cycles 1.3.xlsx','FS 1.3 N max SC 48 cycles.is_cc','C2:C145143');

FSLoadfortyeightcycle=abs(FSLoadfortyeightcycle);
FSLoadMatrixfortyeightcycle=[FSLoadtimefortyeightcycle,FSExtensionfortyeightcycle,a
bs(FSExtensionfortyeightcycle),FSLoadfortyeightcycle,abs(FSLoadfortyeightcycle)];

FSIMP1MHzMatrixfortyeightcycle=[FSIMP1MHzfortyeightcycle;FSPhase1MHzfortyei
ghtcycle;FSIMPtimefortyeightcycle];
FSIMP1MHzMatrixfortyeightcycle=FSIMP1MHzMatrixfortyeightcycle';
```

```
FSIMP1MHzMatrixfortyeightcycle=[FSIMP1MHzMatrixfortyeightcycle(1:(100-
1),:);FSIMP1MHzMatrixfortyeightcycle((100+1):(14603-
1),:);FSIMP1MHzMatrixfortyeightcycle((14603+1):end,:) ];
FSPhase1MHzMatrixfortyeightcycle=[abs(FSPhase1MHzfortyeightcycle-
FSPhase1MHzfortyeightcycle(1));FSIMP1MHzfortyeightcycle;FSIMPtimefortyeightcycl
e];
FSPhase1MHzMatrixfortyeightcycle=FSPhase1MHzMatrixfortyeightcycle';

[FSIMP1MHzinterfortyeightcycle]=analdata5(FSIMP1MHzMatrixfortyeightcycle,FSLoa
dMatrixfortyeightcycle,16.3,46.9);
[FSPhase1MHzinterfortyeightcycle]=analdata5(FSPhase1MHzMatrixfortyeightcycle,FS
LoadMatrixfortyeightcycle,(37),46.9);
FSPhase1MHzinterfortyeightcycle(:,2)=-FSPhase1MHzinterfortyeightcycle(:,2);
FSPhase1MHzinterfortyeightcycle(:,2)=FSPhase1MHzinterfortyeightcycle(:,2)+FSPhase
1MHzfortyeightcycle(1,1);

FSfortyeightavghysterisis=avghyst([FSIMP1MHzinterfortyeightcycle(round(2.*length(F
SIMP1MHzinterfortyeightcycle)/48):round(3.*length(FSIMP1MHzinterfortyeightcycle)/
48),2),FSIMP1MHzinterfortyeightcycle(round(2.*length(FSIMP1MHzinterfortyeightcycl
e)/48):round(3.*length(FSIMP1MHzinterfortyeightcycle)/48),1)],35);
FSerrorreadymatrix=errorreadymatrix(FSIMP1MHzinterfortyeightcycle,40);
FSerrorreadymatrixfinalc=mean(FSerrorreadymatrix(2:2:end,:));
FSerrorreadymatrixfinald=mean(FSerrorreadymatrix(3:2:end,:));
a=250;
b=200;
FSerrc=std(FSerrorreadymatrix(2:2:end,10:a:end));
FSerrd=std(FSerrorreadymatrix(3:2:end,1:b:end));
figure(3);
plot(FSerrorreadymatrix(1,:),FSerrorreadymatrixfinalc,FSerrorreadymatrix(1,:),FSerrorre
adymatrixfinald)
figure(4);
errorbar(FSerrorreadymatrix(1,10:a:end),FSerrorreadymatrixfinalc(10:a:end),1.671.*FSer
rc)
hold on;
errorbar(FSerrorreadymatrix(1,1:b:end),FSerrorreadymatrixfinald(1:b:end),1.671.*FSerrd
)
hold off;

%% PDMS
%One Cycle
% Impedance
PDMSIMPonecycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\ImpedancePDMSSCSEonecycle 04-Aug-2017 12-05-11.txt',',');
PDMSIMPtimeonecycle=PDMSIMPonecycle(1,2:end);
PDMSIMP1MHzonecycle=PDMSIMPonecycle(51,2:end);
```

```
%Phase
PDMSPhaseonecycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\PhasePDMSSCSEonecycle 04-Aug-2017 12-05-12.txt',',');
PDMSPhasetimeonecycle=PDMSPhaseonecycle(1,2:end);
PDMSPhase1MHzonecycle=PDMSPhaseonecycle(51,2:end);
%Load
PDMSLoadtimeonecycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\PDMS Load one cycle.xlsx','PDMS 1.45 mm max SC 1cycles.is_','A2:A123144');
PDMSExtensiononecycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\one
cycle\PDMS Load one cycle.xlsx','PDMS 1.45 mm max SC 1cycles.is_','B2:B123144');
PDMSLoadonecycle=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Compression
tester\Drift test\Force control\one cycle\PDMS Load one cycle.xlsx','PDMS 1.45 mm
max SC 1cycles.is_','C2:C123144');

PDMSLoadonecycle=abs(PDMSLoadonecycle);
PDMSLoadMatrixonecycle=[PDMSLoadtimeonecycle,PDMSExtensiononecycle,abs(PD
MSExtensiononecycle),PDMSLoadonecycle,abs(PDMSLoadonecycle)];

PDMSIMP1MHzMatrixonecycle=[PDMSIMP1MHzonecycle;PDMSPhase1MHzonecycl
e;PDMSIMPtimeonecycle];
PDMSIMP1MHzMatrixonecycle=PDMSIMP1MHzMatrixonecycle';
PDMSPhase1MHzMatrixonecycle=[abs(PDMSPhase1MHzonecycle-
PDMSPhase1MHzonecycle(1));PDMSIMP1MHzonecycle;PDMSIMPtimeonecycle];
PDMSPhase1MHzMatrixonecycle=PDMSPhase1MHzMatrixonecycle';

[PDMSIMP1MHzinteronecycle]=analdata4(PDMSIMP1MHzMatrixonecycle,PDMSLoa
dMatrixonecycle,32.9,34.2);
[PDMSPhase1MHzinteronecycle]=analdata4(PDMSPhase1MHzMatrixonecycle,PDMSL
oadMatrixonecycle,(69.5),34.2);
PDMSPhase1MHzinteronecycle(:,1)=-
PDMSPhase1MHzinteronecycle(:,1)+PDMSPhase1MHzonecycle(1);

PDMSonecycleavghysterisis=avghyst(PDMSIMP1MHzinteronecycle,30)

% 48 Cycle
% Impedance
PDMSIMPfortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\ImpedancePDMSSCSEfortyeightcycle 04-Aug-2017 14-23-37.txt',',');
PDMSIMPtimefortyeightcycle=PDMSIMPfortyeightcycle(1,2:end);
PDMSIMP1MHzfortyeightcycle=PDMSIMPfortyeightcycle(51,2:end);
%Phase
```

```
PDMSPhasefortyeightcycle=importdata('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PhasePDMSSCSEfortyeightcycle 04-Aug-2017 14-23-38.txt',',');
PDMSPhasetimefortyeightcycle=PDMSPhasefortyeightcycle(1,2:end);
PDMSPhase1MHzfortyeightcycle=PDMSPhasefortyeightcycle(51,2:end);
%Load
PDMSLoadtimefortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PDMS Load 48 cycles 1.45.xlsx','PDMS 1.45 mm max SC 48
cycles.i','A2:A144872');
PDMSExtensionfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PDMS Load 48 cycles 1.45.xlsx','PDMS 1.45 mm max SC 48
cycles.i','B2:B144872');
PDMSLoadfortyeightcycle=xlsread('C:\Users\Jacob
Dick\Desktop\Research\Data\Compression tester\Drift test\Force control\48
cycles\PDMS Load 48 cycles 1.45.xlsx','PDMS 1.45 mm max SC 48
cycles.i','C2:C144872');

PDMSLoadfortyeightcycle=abs(PDMSLoadfortyeightcycle);
PDMSLoadMatrixfortyeightcycle=[PDMSLoadtimefortyeightcycle,PDMSExtensionforty
eightcycle,abs(PDMSExtensionfortyeightcycle),PDMSLoadfortyeightcycle,abs(PDMSLo
adfortyeightcycle)];

PDMSIMP1MHzMatrixfortyeightcycle=[PDMSIMP1MHzfortyeightcycle;PDMSPhase1
MHzfortyeightcycle;PDMSIMPtimefortyeightcycle];
PDMSIMP1MHzMatrixfortyeightcycle=PDMSIMP1MHzMatrixfortyeightcycle';
PDMSIMP1MHzMatrixfortyeightcycle=[PDMSIMP1MHzMatrixfortyeightcycle(1:(100-
1),:);PDMSIMP1MHzMatrixfortyeightcycle((100+1):(14603-
1),:);PDMSIMP1MHzMatrixfortyeightcycle((14603+1):end,:) ];
PDMSPhase1MHzMatrixfortyeightcycle=[abs(PDMSPhase1MHzfortyeightcycle-
PDMSPhase1MHzfortyeightcycle(1));PDMSIMP1MHzfortyeightcycle;PDMSIMPtimefo
rtyeightcycle];
PDMSPhase1MHzMatrixfortyeightcycle=PDMSPhase1MHzMatrixfortyeightcycle';

[PDMSIMP1MHzinterfortyeightcycle]=analdata5(PDMSIMP1MHzMatrixfortyeightcycl
e,PDMSLoadMatrixfortyeightcycle,18.9,40);
[PDMSPhase1MHzinterfortyeightcycle]=analdata5(PDMSPhase1MHzMatrixfortyeightc
ycle,PDMSLoadMatrixfortyeightcycle,(50),40);
PDMSPhase1MHzinterfortyeightcycle(:,2)=-PDMSPhase1MHzinterfortyeightcycle(:,2);
PDMSPhase1MHzinterfortyeightcycle(:,2)=PDMSPhase1MHzinterfortyeightcycle(:,2)+
PDMSPhase1MHzfortyeightcycle(1);

PDMSfortyeightavghysterisis=avghyst([PDMSIMP1MHzinterfortyeightcycle(round(2.*l
ength(PDMSIMP1MHzinterfortyeightcycle)/48):round(3.*length(PDMSIMP1MHzinterf
ortyeightcycle)/48),2),FSIMP1MHzinterfortyeightcycle(round(2.*length(PDMSIMP1M
```

```
Hzinterfortyeightcycle)/48):round(3.*length(PDMSIMP1MHzinterfortyeightcycle)/48),1)
],35)
PDMSerrorreadymatrix=errorreadymatrix(PDMSIMP1MHzinterfortyeightcycle,38);
PDMSerrorreadymatrixfinalc=mean(PDMSerrorreadymatrix(2:2:end,:));
PDMSerrorreadymatrixfinald=mean(PDMSerrorreadymatrix(3:2:end,:));
figure(5);
plot(PDMSerrorreadymatrix(1,:),PDMSerrorreadymatrixfinalc,PDMSerrorreadymatrix(1,
:),PDMSerrorreadymatrixfinald)
a=250;
b=200;
PDMSerrc=std(PDMSerrorreadymatrix(2:2:end,10:a:end));
PDMSerrd=std(PDMSerrorreadymatrix(3:2:end,1:b:end));
figure(6);
plot(PDMSerrorreadymatrix(1,:),PDMSerrorreadymatrixfinalc,PDMSerrorreadymatrix(1,
:),PDMSerrorreadymatrixfinald)
figure(7);
errorbar(PDMSerrorreadymatrix(1,10:a:end),PDMSerrorreadymatrixfinalc(10:a:end),1.6
71.*PDMSerrc)
hold on;
errorbar(PDMSerrorreadymatrix(1,1:b:end),PDMSerrorreadymatrixfinald(1:b:end),1.671.
*PDMSerrd)
hold off;
```

Wireless interpolation code

```
%% PVC
clear;
clc;
%SD
% Impedance
PVCIMPSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\ImpedancePVCWSCSEtwentyfourcycle 11-Aug-2017 15-23-37.txt',',');
PVCIMPtimeSD=PVCIMPSD(1,2:end);
PVCIMP1MHzSD=PVCIMPSD(51,2:end);
%Phase
PVCPhaseSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PhasePVCWSCSEtwentyfourcycle 11-Aug-2017 15-23-38.txt',',');
PVCPhasetimeSD=PVCPhaseSD(1,2:end);
PVCPhase1MHzSD=PVCPhaseSD(51,2:end);
%Load
PVCLoadtimeSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC W Load.xlsx','PVC 1.45 mm max WSC 24cycles _1','A2:A56747');
PVCExtensionSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC W Load.xlsx','PVC 1.45 mm max WSC 24cycles _1','B2:B56747');
PVCLoadSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC W Load.xlsx','PVC 1.45 mm max WSC 24cycles _1','C2:C56747');

PVCLoadSD=abs(PVCLoadSD);
```

```
PVCLoadMatrixSD=[PVCLoadtimeSD,PVCExtensionSD,abs(PVCExtensionSD),PVCL
oadSD,abs(PVCLoadSD)];

PVCIMP1MHzMatrixSD=[PVCIMP1MHzSD;PVCPhase1MHzSD;PVCIMPtimeSD];
PVCIMP1MHzMatrixSD=PVCIMP1MHzMatrixSD';
PVCPhase1MHzMatrixSD=[abs(PVCPhase1MHzSD-
PVCPhase1MHzSD(1));PVCIMP1MHzSD;PVCIMPtimeSD];
PVCPhase1MHzMatrixSD=PVCPhase1MHzMatrixSD';
%cleaning up data
[a,b]=findpeaks(PVCIMP1MHzMatrixSD(:,1),'MinPeakHeight',50);
for i=1:length(b)
   PVCIMP1MHzMatrixSD(b(i),1)=PVCIMP1MHzMatrixSD((b(i)-1),1);
end

[PVCIMP1MHzinterSD]=analdata5(PVCIMP1MHzMatrixSD,PVCLoadMatrixSD,46.65
,65);
[PVCPhase1MHzinterSD]=analdata5(PVCPhase1MHzMatrixSD,PVCLoadMatrixSD,(2)
,65);
PVCPhase1MHzinterSD(:,2)=-PVCPhase1MHzinterSD(:,2)+PVCPhase1MHzSD(1,2);

PVCSDavghysterisis=avghyst([PVCPhase1MHzinterSD(round(2.*length(PVCPhase1M
HzinterSD)/24):round(3.*length(PVCPhase1MHzinterSD)/24),2),PVCPhase1MHzinterS
D(round(2.*length(PVCPhase1MHzinterSD)/24):round(3.*length(PVCPhase1MHzinterS
D)/24),1)],60)
sensitivityPVCIMPSD=(max(PVCIMP1MHzinterSD(:,2))-
min(PVCIMP1MHzinterSD(:,2)))/(max(PVCIMP1MHzinterSD(:,1))-35);
sensitivityPVCPHSSD=(max(PVCPhase1MHzinterSD(:,2))-
min(PVCPhase1MHzinterSD(:,2)))/(max(PVCPhase1MHzinterSD(:,1))-35);

% LD
% Impedance
PVCIMPLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\ImpedancePVCWSCSELDtwentyfourcycle 11-Aug-2017 16-18-16.txt',',');
PVCIMPtimeLD=PVCIMPLD(1,2:end);
PVCIMP1MHzLD=PVCIMPLD(51,2:end);
%Phase
PVCPhaseLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PhasePVCWSCSELDtwentyfourcycle 11-Aug-2017 16-18-17.txt',',');
PVCPhasetimeLD=PVCPhaseLD(1,2:end);
PVCPhase1MHzLD=PVCPhaseLD(51,2:end);
%Load
PVCLoadtimeLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC WLD Load.xlsx','PVC 1.45 mm max WSCLD 24 cycles','A2:A144678');
PVCExtensionLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC WLD Load.xlsx','PVC 1.45 mm max WSCLD 24 cycles','B2:B144678');
```

```
PVCLoadLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PVC WLD Load.xlsx','PVC 1.45 mm max WSCLD 24 cycles','C2:C144678');

PVCLoadLD=abs(PVCLoadLD);
PVCLoadMatrixLD=[PVCLoadtimeLD,PVCExtensionLD,abs(PVCExtensionLD),PVCL
oadLD,abs(PVCLoadLD)];

PVCIMP1MHzMatrixLD=[PVCIMP1MHzLD;PVCPhase1MHzLD;PVCIMPtimeLD];
PVCIMP1MHzMatrixLD=PVCIMP1MHzMatrixLD';
% PVCIMP1MHzMatrixfortyeightcycle=[PVCIMP1MHzMatrixfortyeightcycle(1:(100-
1),:);PVCIMP1MHzMatrixfortyeightcycle((100+1):(14603-
1),:);PVCIMP1MHzMatrixfortyeightcycle((14603+1):end,:) ];
PVCPhase1MHzMatrixLD=[abs(PVCPhase1MHzLD-
PVCPhase1MHzLD(1));PVCIMP1MHzLD;PVCIMPtimeLD];
PVCPhase1MHzMatrixLD=PVCPhase1MHzMatrixLD';

%cleaning up data
[a,b]=findpeaks(PVCIMP1MHzMatrixLD(:,1),'MinPeakHeight',70);
for i=1:length(b)
   PVCIMP1MHzMatrixLD(b(i),1)=PVCIMP1MHzMatrixLD((b(i)-1),1);
end

[PVCIMP1MHzinterLD]=analdata5(PVCIMP1MHzMatrixLD,PVCLoadMatrixLD,66.17
,65);
[PVCPhase1MHzinterLD]=analdata5(PVCPhase1MHzMatrixLD,PVCLoadMatrixLD,(.0
3),65);
PVCPhase1MHzinterLD(:,2)=-PVCPhase1MHzinterLD(:,2);
PVCPhase1MHzinterLD(:,2)=PVCPhase1MHzinterLD(:,2)+PVCPhase1MHzLD(1,1);

PVCLDavghysterisis=avghyst([PVCPhase1MHzinterLD(round(2.*length(PVCPhase1M
HzinterLD)/24):round(3.*length(PVCPhase1MHzinterLD)/24),2),PVCPhase1MHzinterL
D(round(2.*length(PVCPhase1MHzinterLD)/24):round(3.*length(PVCPhase1MHzinter
LD)/24),1)],60)
sensitivityPVCIMPLD=(max(PVCIMP1MHzinterLD(:,2))-
min(PVCIMP1MHzinterLD(:,2)))/(max(PVCIMP1MHzinterLD(:,1))-35);
sensitivityPVCPHSLD=(max(PVCPhase1MHzinterLD(:,2))-
min(PVCPhase1MHzinterLD(:,2)))/(max(PVCPhase1MHzinterLD(:,1))-35);

%% FS
% SD
% Impedance
FSIMPSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\ImpedanceFSWSCSEtwentyfourcycle 14-Aug-2017 10-39-23.txt',',');
FSIMPtimeSD=FSIMPSD(1,2:end);
FSIMP1MHzSD=FSIMPSD(51,2:end);
%Phase
```

```
FSPhaseSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PhaseFSWSCSEtwentyfourcycle 14-Aug-2017 10-39-23.txt',',');
FSPhasetimeSD=FSPhaseSD(1,2:end);
FSPhase1MHzSD=FSPhaseSD(51,2:end);
%Load
FSLoadtimeSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\FS W Load.xlsx','FS 1.25 mm max WSCLD 24 cycles ','A2:A69679');
FSExtensionSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\FS W Load.xlsx','FS 1.25 mm max WSCLD 24 cycles ','B2:B69679');
FSLoadSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\FS W
Load.xlsx','FS 1.25 mm max WSCLD 24 cycles ','C2:C69679');

FSLoadSD=abs(FSLoadSD);
FSLoadMatrixSD=[FSLoadtimeSD,FSExtensionSD,abs(FSExtensionSD),FSLoadSD,abs
(FSLoadSD)];

FSIMP1MHzMatrixSD=[FSIMP1MHzSD;FSPhase1MHzSD;FSIMPtimeSD];
FSIMP1MHzMatrixSD=FSIMP1MHzMatrixSD';
FSPhase1MHzMatrixSD=[abs(FSPhase1MHzSD-
FSPhase1MHzSD(1));FSIMP1MHzSD;FSIMPtimeSD];
FSPhase1MHzMatrixSD=FSPhase1MHzMatrixSD';


[FSIMP1MHzinterSD]=analdata5(FSIMP1MHzMatrixSD,FSLoadMatrixSD,46.7,43);
[FSPhase1MHzinterSD]=analdata5(FSPhase1MHzMatrixSD,FSLoadMatrixSD,(3),44);
FSPhase1MHzinterSD(:,2)=-FSPhase1MHzinterSD(:,2)+FSPhase1MHzSD(1,1);

sensitivityFSIMPSD=(max(FSIMP1MHzinterSD(:,2))-
min(FSIMP1MHzinterSD(:,2)))/(max(FSIMP1MHzinterSD(:,1))-5);
sensitivityFSPHSSD=(max(FSPhase1MHzinterSD(:,2))-
min(FSPhase1MHzinterSD(:,2)))/(max(FSPhase1MHzinterSD(:,1))-5);

% LD
% Impedance
FSIMPLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\ImpedanceFSWSCSELDtwentyfourcycle 14-Aug-2017 11-43-35.txt',',');
FSIMPtimeLD=FSIMPLD(1,2:end);
FSIMP1MHzLD=FSIMPLD(51,2:end);
%Phase
FSPhaseLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PhaseFSWSCSELDtwentyfourcycle 14-Aug-2017 11-43-35.txt',',');
FSPhasetimeLD=FSPhaseLD(1,2:end);
FSPhase1MHzLD=FSPhaseLD(51,2:end);
%Load
FSLoadtimeLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\FS WLD Load.xlsx','FS 1.25 mm max WSCLD actual 24 ','A2:A145143');
```

```
FSExtensionLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\FS WLD Load.xlsx','FS 1.25 mm max WSCLD actual 24 ','B2:B145143');
FSLoadLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\FS WLD Load.xlsx','FS 1.25 mm max WSCLD actual 24 ','C2:C145143');

FSLoadLD=abs(FSLoadLD);
FSLoadMatrixLD=[FSLoadtimeLD,FSExtensionLD,abs(FSExtensionLD),FSLoadLD,abs(FSLoadLD)];

FSIMP1MHzMatrixLD=[FSIMP1MHzLD;FSPhase1MHzLD;FSIMPtimeLD];
FSIMP1MHzMatrixLD=FSIMP1MHzMatrixLD';
% FSIMP1MHzMatrixLD=[FSIMP1MHzMatrixLD(1:(100-1),:);FSIMP1MHzMatrixLD((100+1):(14603-1),:);FSIMP1MHzMatrixLD((14603+1):end,:) ];
FSPhase1MHzMatrixLD=[abs(FSPhase1MHzLD-FSPhase1MHzLD(1));FSIMP1MHzLD;FSIMPtimeLD];
FSPhase1MHzMatrixLD=FSPhase1MHzMatrixLD';

[FSIMP1MHzinterLD]=analdata5(FSIMP1MHzMatrixLD,FSLoadMatrixLD,66.22,46.9);
[FSPhase1MHzinterLD]=analdata5(FSPhase1MHzMatrixLD,FSLoadMatrixLD,(.4),46.9);
FSPhase1MHzinterLD(:,2)=-FSPhase1MHzinterLD(:,2);
FSPhase1MHzinterLD(:,2)=FSPhase1MHzinterLD(:,2)+FSPhase1MHzLD(1,1);

sensitivityFSIMPLD=(max(FSIMP1MHzinterLD(:,2))-min(FSIMP1MHzinterLD(:,2)))/(max(FSIMP1MHzinterLD(:,1))-5);
sensitivityFSPHSLD=(max(FSPhase1MHzinterLD(:,2))-min(FSPhase1MHzinterLD(:,2)))/(max(FSPhase1MHzinterLD(:,1))-5);

%% PDMS
% SD
% Impedance
PDMSIMPSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\ImpedancePDMSWSCSEtwentyfourcycle 15-Aug-2017 16-43-29.txt',',');
PDMSIMPtimeSD=PDMSIMPSD(1,2:end);
PDMSIMP1MHzSD=PDMSIMPSD(51,2:end);
%Phase
PDMSPhaseSD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\PhasePDMSWSCSEtwentyfourcycle 15-Aug-2017 16-43-30.txt',',');
PDMSPhasetimeSD=PDMSPhaseSD(1,2:end);
PDMSPhase1MHzSD=PDMSPhaseSD(51,2:end);
%Load
PDMSLoadtimeSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\PDMS W Load.xlsx','PDMS 1.3 mm max WSC actual 24 c','A2:A123144');
```

```
PDMSExtensionSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PDMS W Load.xlsx','PDMS 1.3 mm max WSC actual 24 c','B2:B123144');
PDMSLoadSD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PDMS W Load.xlsx','PDMS 1.3 mm max WSC actual 24 c','C2:C123144');

PDMSLoadSD=abs(PDMSLoadSD);
PDMSLoadMatrixSD=[PDMSLoadtimeSD,PDMSExtensionSD,abs(PDMSExtensionSD
),PDMSLoadSD,abs(PDMSLoadSD)];

PDMSIMP1MHzMatrixSD=[PDMSIMP1MHzSD;PDMSPhase1MHzSD;PDMSIMPtim
eSD];
PDMSIMP1MHzMatrixSD=PDMSIMP1MHzMatrixSD';
PDMSPhase1MHzMatrixSD=[abs(PDMSPhase1MHzSD-
PDMSPhase1MHzSD(1));PDMSIMP1MHzSD;PDMSIMPtimeSD];
PDMSPhase1MHzMatrixSD=PDMSPhase1MHzMatrixSD';


[PDMSIMP1MHzinterSD]=analdata5(PDMSIMP1MHzMatrixSD,PDMSLoadMatrixSD,
47.5,45);
[PDMSPhase1MHzinterSD]=analdata5(PDMSPhase1MHzMatrixSD,PDMSLoadMatrix
SD,(5),45);
PDMSPhase1MHzinterSD(:,2)=-
PDMSPhase1MHzinterSD(:,2)+PDMSPhase1MHzSD(1);

sensitivityPDMSIMPSD=(max(PDMSIMP1MHzinterSD(:,2))-
min(PDMSIMP1MHzinterSD(:,2)))/(max(PDMSIMP1MHzinterSD(:,1))-5);
sensitivityPDMSPHSSD=(max(PDMSPhase1MHzinterSD(:,2))-
min(PDMSPhase1MHzinterSD(:,2)))/(max(PDMSPhase1MHzinterSD(:,1))-5);

% LD
% Impedance
PDMSIMPLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\ImpedancePDMSWSCSELDtwentyfourcycle 15-Aug-2017 17-45-22.txt',',');
PDMSIMPtimeLD=PDMSIMPLD(1,2:end);
PDMSIMP1MHzLD=PDMSIMPLD(51,2:end);
%Phase
PDMSPhaseLD=importdata('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PhasePDMSWSCSELDtwentyfourcycle 15-Aug-2017 17-45-22.txt',',');
PDMSPhasetimeLD=PDMSPhaseLD(1,2:end);
PDMSPhase1MHzLD=PDMSPhaseLD(51,2:end);
%Load
PDMSLoadtimeLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PDMS WLD Load.xlsx','PDMS 1.3 mm max WSCLD actual 24','A2:A144872');
PDMSExtensionLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless
Testing\PDMS WLD Load.xlsx','PDMS 1.3 mm max WSCLD actual 24','B2:B144872');
```

```
PDMSLoadLD=xlsread('C:\Users\Jacob Dick\Desktop\Research\Data\Wireless Testing\PDMS WLD Load.xlsx','PDMS 1.3 mm max WSCLD actual 24','C2:C144872');

PDMSLoadLD=abs(PDMSLoadLD);
PDMSLoadMatrixLD=[PDMSLoadtimeLD,PDMSExtensionLD,abs(PDMSExtensionLD),PDMSLoadLD,abs(PDMSLoadLD)];

PDMSIMP1MHzMatrixLD=[PDMSIMP1MHzLD;PDMSPhase1MHzLD;PDMSIMPtimeLD];
PDMSIMP1MHzMatrixLD=PDMSIMP1MHzMatrixLD';
% PDMSIMP1MHzMatrixLD=[PDMSIMP1MHzMatrixLD(1:(100-1),:);PDMSIMP1MHzMatrixLD((100+1):(14603-1),:);PDMSIMP1MHzMatrixLD((14603+1):end,:) ];
PDMSPhase1MHzMatrixLD=[abs(PDMSPhase1MHzLD-PDMSPhase1MHzLD(1));PDMSIMP1MHzLD;PDMSIMPtimeLD];
PDMSPhase1MHzMatrixLD=PDMSPhase1MHzMatrixLD';

[PDMSIMP1MHzinterLD]=analdata5(PDMSIMP1MHzMatrixLD,PDMSLoadMatrixLD,65.65,45);
[PDMSPhase1MHzinterLD]=analdata5(PDMSPhase1MHzMatrixLD,PDMSLoadMatrixLD,(.6),45);
PDMSPhase1MHzinterLD(:,2)=-PDMSPhase1MHzinterLD(:,2);
PDMSPhase1MHzinterLD(:,2)=PDMSPhase1MHzinterLD(:,2)+PDMSPhase1MHzLD(1);

sensitivityPDMSIMPLD=(max(PDMSIMP1MHzinterLD(:,2))-min(PDMSIMP1MHzinterLD(:,2)))/(max(PDMSIMP1MHzinterLD(:,1))-5);
sensitivityPDMSPHSLD=(max(PDMSPhase1MHzinterLD(:,2))-min(PDMSPhase1MHzinterLD(:,2)))/(max(PDMSPhase1MHzinterLD(:,1))-5);
```

References

[1] C. Majidi, *Soft Robot.* **2014**, *1*, 5.
[2] Y.-L. Park, C. Majidi, R. Kramer, P. Bérard, R. J. Wood, *J. Micromechanics Microengineering* **2010**, *20*, 125029.
[3] Yong-Lae Park, Bor-Rong Chen, R. J. Wood, *IEEE Sens. J.* **2012**, *12*, 2711.
[4] J.-B. Chossat, Y.-L. Park, R. J. Wood, V. Duchaine, *IEEE Sens. J.* **2013**, *13*, 3405.
[5] F. Ilievski, A. D. Mazzeo, R. F. Shepherd, X. Chen, G. M. Whitesides, *Angew. Chem.* **2011**, *123*, 1930.
[6] J. Xie, Q. Chen, P. Suresh, S. Roy, J. F. White, A. D. Mazzeo, *Proc. Natl. Acad. Sci.* **2017**, *114*, 5119.
[7] Y. Mengüç, Y.-L. Park, E. Martinez-Villalpando, P. Aubin, M. Zisook, L. Stirling, R. J. Wood, C. J. Walsh, In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*; IEEE, 2013; pp. 5309–5316.
[8] C. J. Walsh, D. Paluska, K. Pasch, W. Grand, A. Valiente, H. Herr, In *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*; IEEE, 2006; pp. 3485–3491.
[9] P. Guay, S. Gorgutsa, S. LaRochelle, Y. Messaddeq, *Sensors* **2017**, *17*, 1050.
[10] M. He, K. Zhang, G. Chen, J. Tian, B. Su, *ACS Appl. Mater. Interfaces* **2017**, *9*, 16466.
[11] S. Zhu, J.-H. So, R. Mays, S. Desai, W. R. Barnes, B. Pourdeyhimi, M. D. Dickey, *Adv. Funct. Mater.* **2013**, *23*, 2308.
[12] S. Xu, Y. Zhang, L. Jia, K. E. Mathewson, K.-I. Jang, J. Kim, H. Fu, X. Huang, P. Chava, R. Wang, John A. Rogers, *Science* **2014**, *344*, 70.
[13] S. Yang, Y.-C. Chen, L. Nicolini, P. Pasupathy, J. Sacks, B. Su, R. Yang, D. Sanchez, Y.-F. Chang, P. Wang, D. Schnyer, D. Neikirk, N. Lu, *Adv. Mater.* **2015**, *27*, 6423.
[14] L. Stirling, C.-H. Yu, J. Miller, E. Hawkes, R. Wood, E. Goldfield, R. Nagpal, *J. Mater. Eng. Perform.* **2011**, *20*, 658.
[15] J. Rogers, Y. Huang, *Proc. Natl. Acad. Sci.* **2009**, *106*, 16889.
[16] D.-H. Kim, ong-H. Ahn, W. M. Choi, H.-S. Kim, T.-H. Kim, J. Song, Y. Huang, Z. Liu, C. Lu, J. Rogers, *Science* **2008**, *320*, 504.
[17] N. Lu, C. Lu, S. Yang, J. Rogers, *Adv. Funct. Mater.* **2012**, *22*, 4044.
[18] A. Frutiger, J. T. Muth, D. M. Vogt, Y. Mengüç, A. Campo, A. D. Valentine, C. J. Walsh, J. A. Lewis, *Adv. Mater.* **2015**, *27*, 2440.
[19] J. W. Boley, E. L. White, G. T.-C. Chiu, R. K. Kramer, *Adv. Funct. Mater.* **2014**, *24*, 3501.
[20] A. Deivasigamani, A. Daliri, C. H. Wang, S. John, *Mod. Appl. Sci.* **2013**, *7*.
[21] L. String, P. PAU, *Ariel* **2002**, *128*, 77.
[22] Y. Jia, K. Sun, Tomizuka, M.; Yun, C.-B.; Giurgiutiu, V., Eds.; 2006; p. 61740Z.
[23] E. L. Tan, B. D. Pereles, R. Shao, J. Ong, K. G. Ong, *Smart Mater. Struct.* **2008**, *17*, 025015.
[24] P.-J. Chen, S. Saati, R. Varma, M. S. Humayun, Y.-C. Tai, *J. Microelectromechanical Syst.* **2010**, *19*, 721.
[25] M. A. Fonseca, M. G. Allen, J. Kroh, J. White, In *Tech. Dig. Solid-State Sensor, Actuator, and Microsystems Workshop (Hilton Head 2006)*; 2006; pp. 37–42.
[26] S. H. Jeong, Z. Wu, In *Micro Electro Mechanical Systems (MEMS), 2015 28th IEEE International Conference on*; IEEE, 2015; pp. 1137–1140.

[27] M. D. Dickey, R. C. Chiechi, R. J. Larsen, E. A. Weiss, D. A. Weitz, G. M. Whitesides, *Adv. Funct. Mater.* **2008**, *18*, 1097.
[28] W. V. Titow, *PVC Plastics Properties,Processing, and Applications*; Elsevier Science Publishers LTD: New York, NY, 1990.
[29] O. R. Pierce, G. W. Holbrook, O. K. Johannson, J. C. Saylor, E. D. Brown, *Ind. Eng. Chem.* **1960**, *52*, 783.
[30] J. C. Lötters, W. Olthuis, P. H. Veltink, P. Bergveld, *J. Micromechanics Microengineering* **1997**, *7*, 145.
[31] M. A. Meyers, K. K. Chawla, *Mechanical behavior of materials*; 2nd ed.; Cambridge University Press: Cambridge ; New York, 2009.
[32] C. Koo, B. E. LeBlanc, M. Kelley, H. E. Fitzgerald, G. H. Huff, A. Han, *J. Microelectromechanical Syst.* **2015**, *24*, 1069.
[33] W. L. Robb, *Ann. N. Y. Acad. Sci.* **1968**, *146*, 119.