Honors Capstone Final Report

Capacitance Bridge Project

Jack Barlow

Introduction

Since the late twentieth century, the subject of two-dimensional materials has become an important and rapidly-growing field in the subject of condensed matter physics. This initially followed from the desire to make field-effect transistors even thinner to maintain the pace of Moore's law, which is the observation that the number of transistors in integrated circuits doubles every two years. The study of ultrathin semiconductors led to the isolation of the first atomically-thin layers of graphene in 2004 by Dr. Andre Geim and Dr. Konstantin Novoselov of the University of Manchester, and since other two-dimensional semiconducting materials have been isolated and studied.

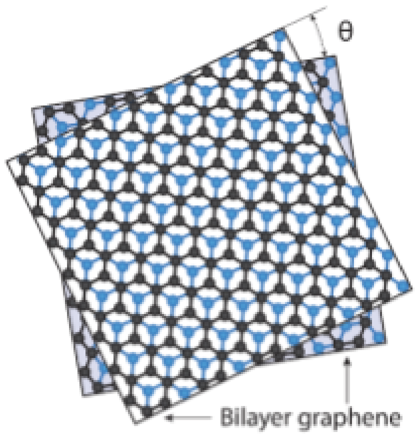


Figure 1: Bilayer graphene, one of the first two-dimensional materials isolated in a lab. It was shown that the phase of graphene bilayer systems can be altered by rotating the sheets with respect to one another by an angle θ.

When two of these atomically-thin layers are stacked on top of one another (or if one of them is placed above a thin metallic plate) they form a parallel plate capacitor, much like when two plates of metal are stacked with a dielectric layer placed between them. The capacitance of the resulting device is defined by the amount of electric charge stored on the device per unit voltage difference applied, i.e. by the relation $C=Q/V$ where Q is the total charge and V is the voltage difference across the device. The charge separation forms when you apply a voltage difference to a circuit containing these separate plates. The voltage difference causes the electrons to move, but since the plates of the capacitor are not touching save for an insulating dielectric between them, the electrons just accumulate on one of the plates and then the plates have different amounts of charge. Thus, a capacitor is formed whenever you apply a voltage to two separate plates of material that allow for the motion of electrons. The key difference between these semiconducting bilayers (dual layers of atomically-thin materials) and a standard capacitor that you would find in an analog device is the way that electrons move through them, which has a major effect on the capacitance of the resulting device.

In a standard parallel plate capacitor, the plates are made of metal which has ideal or near-ideal conducting properties, so the electrons are free to move from the plates and through the wires and collect on them regardless of the energy of the electrons. In this case, if the plates are large enough with respect to the distance between the plates, the capacitance of the device can be easily determined using the equation $C = \varepsilon_0 A/d$, where A is the area of the parallel

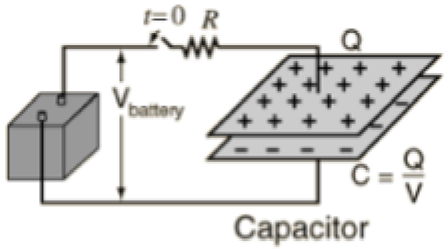


Figure 2: an RC circuit demonstrating charge buildup on standard capacitor plates.

plates, d is the distance between the plates, and ε_0 is the permittivity of free space. This is what we call the geometric capacitance, and it is different for different capacitor configurations.

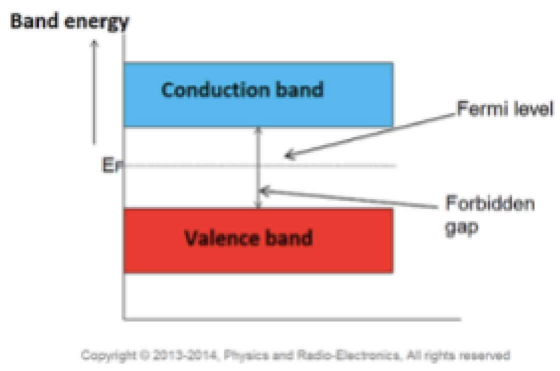

Band energy
Conduction band
Fermi level
E_F
Forbidden gap
Valence band
Copyright © 2013-2014, Physics and Radio-Electronics, All rights reserved

Figure 3: an illustration of the band gap of an intrinsic semiconductor. The Fermi level determines whether electrons can exist in the conduction band at 0K, but with thermal excitation some electrons can be promoted to the conduction band.

In the case when one or both of the plates are semiconductors, however, the layers act as a hybrid between metal and insulators. This means that the only electrons that are allowed to move freely are the electrons that can overcome the band gap of the semiconducting material. This band gap is a range of energies that the electrons cannot occupy due to the quantum effects of the material, i.e. the electrons can exist in the energies below the band in an insulating state or, if enough energy is added to the system, the electrons can overcome the band gap to reach a conducting state and move freely through the circuit. The number of electrons that can be promoted to the conduction band is also limited by the system energy and a quantity called the Fermi level or electrochemical potential of the system. This limitation on the number of electrons that can occupy the conduction band follows from the Pauli exclusion principle, which states that no two fermions (particles with half-integer spin values, which includes electrons) can occupy both the same energy and spin states in a confined space, thus limiting the amount of charge that can accumulate on the plates of the capacitor. Thus, by changing the energy of the system to allow more electrons to enter the conduction band, we can change the capacitance of the device.

Additionally, when the higher-energy electrons gather on one plate and leave the lower-energy electrons behind, they generate an additional potential energy difference which alters the capacitance, generating what we call the quantum capacitance. This capacitance is much greater than the geometric capacitance in these kinds of two-dimensional nanoscale devices. Not to mention that it can also theoretically lower the capacitance to a negative value, which is the subject of some research interest.

The quantum capacitance is proportional to the number of electrons that are available to promote to the conducting band at a given energy. There is a name for this quantity - we call it the "density of states", or the number of states a system can exist in as a function of energy. It is exactly proportional to the quantum capacitance by the relation $C_Q = e^2 D(E)$, where e is the elementary unit of charge and $D(E)$ is the density of states. The density of states is important whenever we consider the thermodynamic and statistical properties of the material, since it is often used alongside the Fermi-Dirac distribution as a weight to compute statistical averages of the material properties.

The two most important quantities that rose as a consequence of analyzing a capacitor made of at least one semiconductor layer are the density of states and the quantum capacitance, which just so happen are related by a constant of proportionality that is well-known. We also noted that we can change the capacitance of the system by altering the system energy, and that the quantum capacitance is the most important contributing quantity to the overall capacitance in these low-dimensional semiconducting systems. Thus, if there was a way to measure the capacitance of the device, and if we measured the capacitance for a range of energies, we could not only find the quantum capacitance spectrum of the device (a useful quantity on its own), but we could also find the density of states of the device, which gives a lot of extra insight to the properties of that material as well.

Here enters the capacitance bridge. A capacitance bridge is a circuit that contains one unknown capacitor and is very sensitive to changes in the capacitance of the unknown, allowing us to easily determine the capacitance of the sample. We can then apply a gate voltage to the device to alter the system energy. With this capacitance bridge circuit and a device to apply a gate voltage, we have a way to vary the energy of the device and measure its capacitance as a function of the system energy, which in turn yields the capacitance spectrum and the density of states of the material. The capacitance bridge circuit itself looks like:

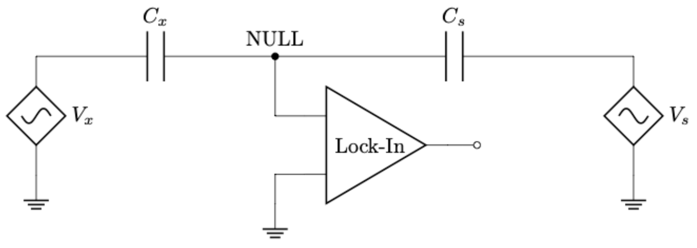


Figure 4: Ideal Capacitance Bridge Circuit

It is composed of a semiconductor device with unknown capacitance C_X, a standard capacitor of known capacitance C_S, a dual-output signal generator controlling the voltage sources V_X and V_S, and a lock-in amplifier to measure the voltage at the capacitance bridge node. We can alter the energy of the material sample by applying a gate voltage to it using a DC voltage source. By holding one voltage source V_S constant and adjusting the phase and amplitude of the other voltage source V_X using some kind of feedback loop, we can find the phase and amplitude values that cause the voltage at the point labeled "NULL" to read zero. Under ideal conditions, when the voltage at "NULL" is zero, Kirchoff's Junction Law reduces to $V_X C_X = V_S C_S$. Then, since we control V_X and V_S and C_S is known, we can solve for the device capacitance C_X. Performing this process for a range of applied voltage values from the DC voltage source yields the capacitance spectrum and density of states of the semiconductor device.

Outline of Problems Addressed

The problem of measuring the capacitance spectrum of a semiconductor device has been around for a while, and there is already specialized equipment to do this automatically. Fully functional commercial capacitance bridges exist that can measure the capacitance spectrum of a device with good resolution. However, they are generally very expensive and we wanted a way to do this that was both cost-effective and gave decent resolution for high-frequency excitations. Thus, in order to record the capacitance spectrum of a semiconductor device for our purposes, we needed the following elements:

1. A programmable signal generator, to apply in-sync sinusoidal excitations to both the device and the standard capacitor
2. A programmable DC voltage source to apply a gate voltage to the device
3. A lock-in amplifier to measure the signal at the "NULL" point of the capacitance bridge corresponding to the excitations

The lab was already equipped with lock-in amplifiers, so this was covered. We intended to build the signal generator and the DC voltage source from scratch specifically for this application, but the home-made signal generator had some issues and we ended up using one that was in the lab.

The next step was to complete the necessary programming to configure these components and have them work in coordination with one another to balance a capacitance bridge circuit and export the phase and amplitude values that balance the bridge and determine the capacitance values for each applied DC voltage value. The function that performs the actual loop could be done in any language, but the interface ultimately needed to be compatible with LabVIEW, the primary data acquisition program used in our lab.

Once we had a program that could balance the bridge, we wanted to be able to test our code by performing a frequency sweep measurement with our capacitance bridge. Most capacitors have a resistive element that can be represented using a parallel resistor, and the leads from the capacitor to the balance point often have some resistance that can be represented by a series resistor. The resulting "real" system looks like:

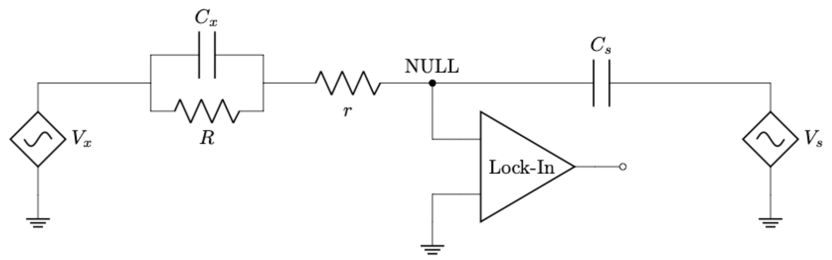


Figure 5: Real capacitance bridge approximation

When the circuit is balanced and the frequency of the voltage sources is near $\omega = 0$, the tangent of the phase of the voltage source V_X is proportional to ω^{-1}. Similarly, when the frequency ω is significantly higher than $\omega = 0$, the tangent of the phase of the voltage source V_X is proportional to ω. Thus, if we sweep through a large frequency range, we should see a transition in the frequency dependence of the tangent of the phase of V_X from ω^{-1} to ω as the frequency is increased.

Methods

Initially, the tasks we had to complete were as follows:

1. Build the signal generator
2. Build the DC voltage generator
3. Write the program to automatically balance the bridge
4. Adapt the program to work in a LabVIEW UI

The programmable DC voltage source was built by a former REU student in our lab according to the specifications of the openDACS project listed in the references. It uses one AD5764 evaluation board and one AD7734 evaluation board controlled by an Arduino Due with a specialized shield soldered on top to output voltage values between +10V and -10V.

The programmable signal generator proved to be much more difficult. The first iteration was intended to be a spin-off of another openDACS project for a AC signal generator, also listed in the references. This configuration is similar to that of the DC voltage source - it uses two AD9854 evaluation boards controlled by an Arduino Due with another (different) specialized shield soldered on top to output a wide range of sinusoidal and square wave AC signals at frequencies of up to 2 MHz. We modified this configuration by using the same evaluation boards, but used a version of them that had a USB interface instead of a serial interface for communicating with the Arduino Due.

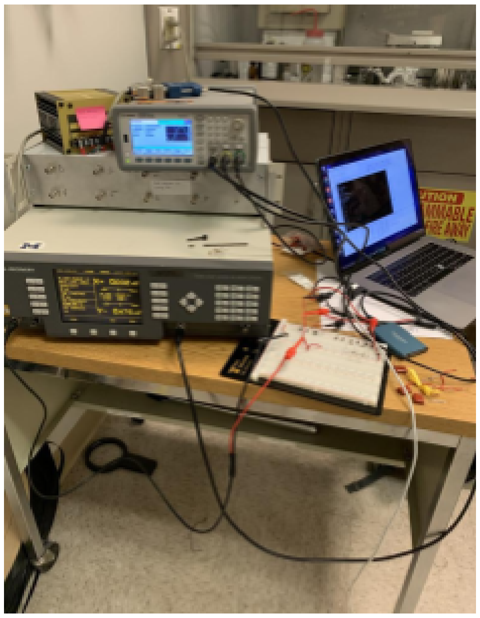


Figure 6: Equipment setup.

The USB signal generators did not come with any information regarding programming them, only some software that could control the board output. Since we needed to be able to program the output automatically rather than inputting values to software, much of my time was spent decoding the information that was sent by the software over USB to the boards. Once I was able to decode many of the bits in the bitstrings for various board operations, we were able to use a Python program to send the same bitstrings to the boards and remotely control the boards in this manner.

Once we were able to program these boards to yield the desired output, we needed to assemble the signal generator that synchronized the output from two boards simultaneously. Each board would need a synchronized reference signal operating at the same high frequency and would need to have roughly the same length of cable connecting them to this reference to eliminate any reasonable source of desynchronization. In the original configuration presented by openDACS,

this is done by placing a reference oscillator on the customized shield that gets soldered onto the Arduino Due, which is then wired to the external reference input of the two evaluation boards. Since we were working to circumvent the Arduino, we had to formulate an alternate solution. Our idea was to have the square wave output of one board act as the reference signal for the other board, which would have theoretically had them both operating with respect to the same reference and should have synchronized their output.
However, this did not result in synchronized output. The reason this did not work was because the architecture of the evaluation boards requires that the reference input signal should be approximately 30MHz, and needs to be told if this value changes but still only works correctly if the oscillator is on the order of 10-20MHz minimum. This exceeded the maximum output frequency of the board we intended to use as reference by an entire order of magnitude, and even when we tried to use the maximum output frequency it still did not result in steady, synchronized output. The next idea was to try and introduce an external oscillator, but such external oscillators are generally quite expensive on their own and eventually this complication resulted in the design being scrapped.

For the second iteration of the signal generator, we tried to use the configuration that was specified initially on the openDACS site using the two evaluation boards and an Arduino Due to control them. The Arduino is prepared with a shield soldered on top that contains some extra functionalities, including a reference oscillator crystal that was discussed earlier as well as some wiring to support LED indicators for the status of the output. The shield also contained all of the pins that would connect the boards' serial output pins to the Arduino. This configuration is also equipped with a USB isolator between the USB input and the Arduino's USB input, which prevents noise from interacting with the commands sent by the computer to the signal generator.

However, this configuration also ran into several problems. Firstly, the onset of the ongoing COVID-19 pandemic delayed much of the progress on constructing the signal generator. Aside from this, there were also some technical issues. Right off the bat, we were unable to send commands to the Arduino after constructing the signal generator. We solved this by circumventing the USB isolator by sending commands directly to the Arduino, which indicated that the USB isolator was not functioning properly. We tried to move on without the USB isolator, thinking that we would re-introduce it to the signal generator after fixing other bugs. Then, the Arduino program that we were provided with to instruct the Arduino's operation did not compile. This was due to some incompatibilities with the compiler and the program, and required extensive research to rectify. After we were finally able to get the code to compile, the signal generator still provided no output. This could have been for many reasons - for example, the wiring from the evaluation boards to the Arduino is very complex and lends itself easily to errors which would cause the output to not work properly. Checking and re-checking the connections provided no further insight, and we were able to verify that the Arduino was communicating properly with the boards and was functioning properly. Eventually we were only able to obtain DC signals from each output indicating that power was flowing through the boards, but none of the desired alternating signal output.

After much deliberation, we decided to place this configuration on the backburner to proceed with completing the capacitance bridge and the program for balancing it. To substitute the failed signal generators, we used a KeySight 33500B programmable waveform generator that was present in the lab. This is capable of performing much of what the signal generators we were working on are able to do, and most importantly it has two output signals that can be synchronized and that have independent phase and amplitude controls for each output. It also comes equipped with a reference signal output that is quite useful for setting the reference frequency of the lock-in amplifier.

Once all of the components were ready, we constructed the capacitance bridge circuit using a breadboard and prepared to test our program. We decided to use Python as our primary language, since it is quite flexible and as such lends itself very easily to remote programming of laboratory equipment. The package PyVISA allows for easy communication with VISA devices over GPIB connection (like the waveform generator and lock-in amplifier we are using) and the package PySerial allows for easy communication with other serial devices (like the house-made DC voltage source).

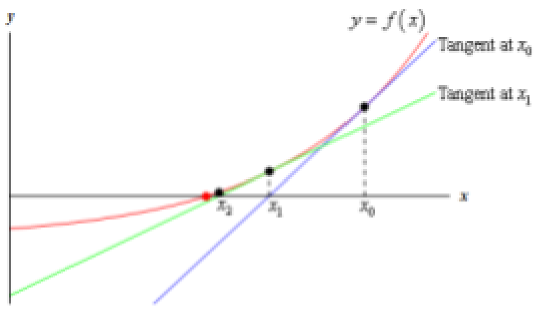


Figure 7: An Illustration of Newton's method.

Once we were able to establish the means of communicating with the equipment, we had to determine a method of balancing the capacitance bridge. Since the bridge voltage reads zero when the desired conditions are met, our heads initially turned to root-finding algorithms to help us balance the bridge. The first one we used is called the secant method, a discretized version of Newton's method for finding the roots of functions. It starts with an initial guess of the root of the function x_0. Then, you use the function $f(x)$ and its derivative $f'(x)$ to make a more educated guess $x_1 = x_0 - f(x_0)/f'(x_0)$, and you repeat this process of making more educated guesses of the location of the root until the change of the value after each iteration satisfies a desired level of accuracy. To discretize this, we need to change the function derivative to a finite difference using the definition of the derivative:

$$f'(x) \;=\; \lim_{\varepsilon \to 0} \left(\frac{f(x+\varepsilon) - f(x)}{\varepsilon}\right) \;\rightarrow\; f'(x) \;\approx\; \frac{f(x+\Delta x) - f(x)}{\Delta x}$$

Here, $f(x)$ corresponds to the voltage at the balance point read by the lock-in amplifier and x corresponds to a given phase and then amplitude of the excitation signal. It also converges to a zero incredibly quickly, having quadratic convergence. However, this method does not place any constraints on the guesses for phase and amplitude of the excitation, even though the phase must be an angle on the unit circle (i.e. between 0º and 360º) and the amplitude must be a positive number. The phase constraint can be easily solved by applying a modulo operator to the output of the algorithm, but the occasional prediction of negative amplitudes could not be reconciled.

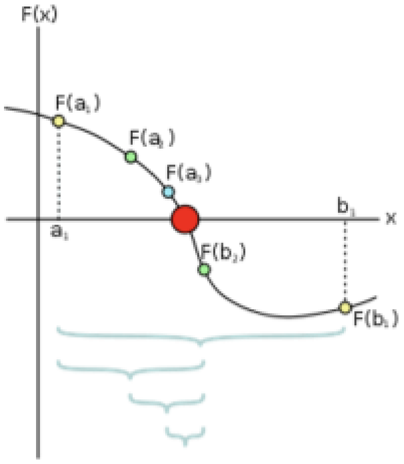


Figure 8: Illustration of the bisection method.

We then moved on to a binary search algorithm. This is rooted in binary search tree data structures, but can be applied to function roots as well. The idea is that if a function $f(x)$ switches signs on the interval $x \in [a,b]$ then there is a value of x for which the function $f(x)$ is equal to zero in that interval. You can then check the value of $f(x)$ at the midpoint m between a and b. Since $f(m)$ will be one of positive, negative, or zero, we have either reduced our interval on which the zero exists to $[a, m]$ or $[m, b]$, or we have found our zero. We can repeat this bisection of the interval until we get a change in the midpoint value that meets a desired level of accuracy. This has already been much more successful than Newton's method, since we can place constraints on the amplitude and phase by limiting the search interval $[a,b]$. However, this algorithm converges to a zero slower than Newton's method does, since it halves the search area on each iteration and thus converges linearly instead of quadratically. While this method still converges relatively quickly, since even linear convergence is quite fast on most computers, it may affect the speed of the program in the future.

In the system, we use the binary search algorithm to determine the phase of V_X which moved all of the signal to the X channel of the lock-in amplifier, i.e to find the phase of V_X for which the Y channel reads 0 V. Then, we use another binary search algorithm to determine the amplitude of V_X for which the X channel then reads zero. The real part of the equation, i.e. $V_X \cos(\varphi)$, can then be substituted into the expression $V_X C_X = V_S C_S$ for V_X to determine the capacitance of the device.

Results

All the following capacitance measurements were performed with V_S = 10mV and a 0° phase offset at a frequency of 10kHz. We ran three test trials to identify the capacitance of various capacitors with the following results:

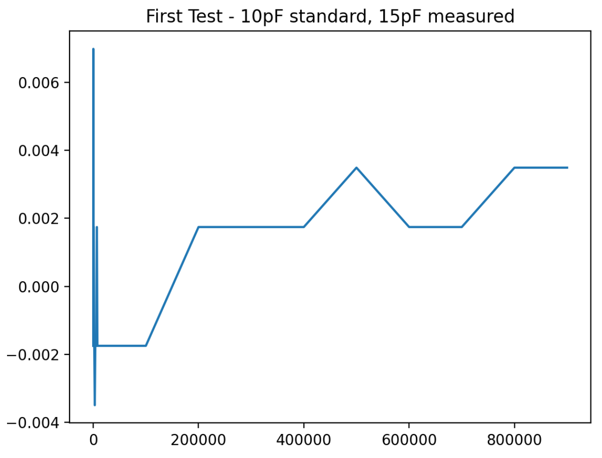
	C_S	C_X Predicted	Phase φ	Amplitude	C_X Measured
Trial 1	48pF	68pF	192.1°	0.00612 V	80.21pF
Trial 2	48pF	101pF	197.6°	0.004524 V	111.3pF
Trial 3	48pF	123pF	201.3°	0.003618 V	142.4pF

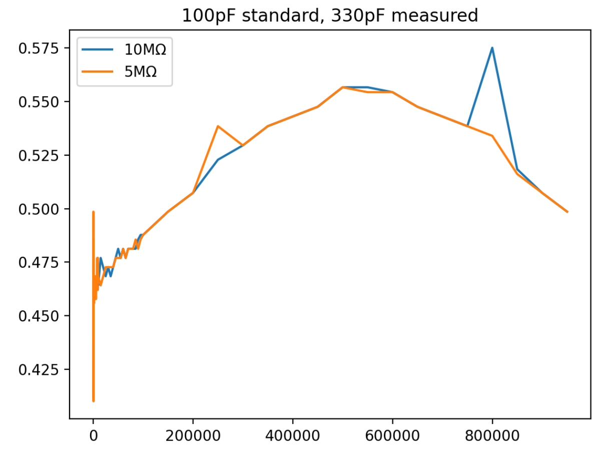
We then repeated this for different standard capacitors and repeated five trials for each configuration:

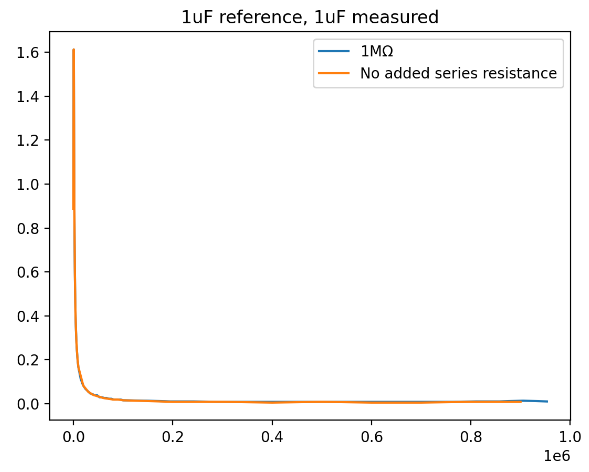
	C_S	C_X Predicted	Phase φ	Amplitude	C_X Measured	% Error	Average	Std Dev
Trial 1	22pF	68pF	204.1º	0.002856 V	84.39pF	19.4%	83.19pF	0.69pF
			203.7º	0.002904 V	82.99pF	17.8%		
			204.1º	0.002904 V	83.08pF	18.1%		
			203.7º	0.002892 V	83.07pF	18.1%		
			203.7º	0.002904 V	82.97pF	17.8%		
Trial 2	4.7pF	68pF	211.8º	0.001006 V	54.97pF	23.7%	55.11pF	0.09pF
			212.1º	0.001006 V	55.15pF	23.3%		
			212.1º	0.001006 V	55.15pF	23.3%		
			212.1º	0.001006 V	55.15pF	23.3%		
Trial 3	6.8pF	68pF	210.9º	0.001066 V	73.34pF	8.5%	74.77pF	0.53pF
			211.3º	0.001054 V	75.51pF	9.9%		
			210.9º	0.001066 V	74.34pF	8.5%		
			211.1º	0.001066 V	74.50pF	8.7%		
			210.9º	0.001054 V	75.19pF	9.6%		
Trial 4	68pF	47pF	157.4º	0.02243 V	32.84pF	46.2%	32.95pF	0.27pF
			157.4º	0.02237 V	32.93pF	45.8%		
			157.9º	0.02209 V	33.22pF	44.5%		
			157.9º	0.02212 V	33.18pF	44.7%		
			157.1º	0.02267 V	32.56pF	47.4%		
Trial 5	68pF	100pF	191.6º	0.00628 V	110.5pF	9.5%	110.3pF	0.42pF
			191.3º	0.00626 V	110.7pF	9.7%		

			191.6º 191.3º 191.3º	0.00633 V 0.00629 V 0.00628 V	109.6pF 110.2pF 110.4pF	8.8% 9.3% 9.4%		
Trial 6	68pF	330pF	205.1º 205.3º 205.1º 205.3º 205.1º	0.002578 V 0.002578 V 0.002578 V 0.002566 V 0.002578 V	291.2pF 291.7pF 291.2pF 293.1pF 291.2pF	13.3% 13.1% 13.3% 12.6% 13.3%	291.7pF	0.80pF
Trial 7	100p F	100pF	180.5º 180.7º 180.7º 180.7º 180.7º	0.00978 V 0.00976 V 0.00974 V 0.00973 V 0.00972 V	102.2pF 102.5pF 102.7pF 102.8pF 102.9pF	2.2% 2.4% 2.6% 2.7% 2.8%	102.6pF	0.25pF
Trial 8	100p F	150pF	186.8º 187.1º 186.9º 186.9º 187.1º	0.00759 V 0.00759 V 0.00763 V 0.00761 V 0.00758 V	132.7pF 132.8pF 132.0pF 132.4pF 132.9pF	13.0% 13.0% 13.6% 13.3% 12.8%	132.6pF	0.37pF
Trial 9	100p F	220pF	193.4º 193.5º 193.4º 193.4º 193.2º	0.00572 V 0.00568 V 0.00568 V 0.00571 V 0.00568 V	179.7pF 181.1pF 181.0pF 180.0pF 180.8pF	22.4% 21.5% 21.6% 22.2% 21.7%	180.5pF	0.61pF
Trial 10	100p F	330pF	200.9º 200.9º 200.9º 201.1º 200.9º	0.003666 V 0.003678 V 0.003690 V 0.003654 V 0.003660 V	292.0pF 291.0pF 290.1pF 293.pF 292.5pF	13.0% 13.4% 13.8% 12.5% 13.8%	291.7pF	1.26pF

Our frequency sweeps yielded the following plots:







Discussion & Conclusions

The capacitance measurements seemed to demonstrate a wide variety of error values, with no clear correlation between the reference capacitor and the measured capacitor. The capacitors we used were only rated for +/-2.5pF or +/- 5% of their listed value, depending on the material they were made of and their capacitance rating, so some error is to be expected. Additionally, the accuracy of the phase was only determined to the tenths place, so the Y-channel of the lock-in amplifier never went totally to zero and thus the bridge was never totally balanced, which will also lead to some error in the measurements. However, most of the measurements had very low standard deviations on the order of the rating of the capacitors, which means that the measurements are fairly precise if not accurate. Also, the high error measured in Trial 2 was due to the fact that the standard capacitor was less than one-tenth of the value of the measured capacitor, and the amplitude range we chose to sweep was only one-tenth of V_S to ten times V_S, meaning the correct amplitude was never reached but it correctly saturated the amplitude value down to the lower bound of the constraint. The much higher error in the measurements of trial 4 is likely due to an issue with the 48pF capacitor we were measuring, since they were soldered on to header pins for placement on the breadboard which left a lot of room for the burning of the capacitor or some other related source of error.

Additionally, we did not quite see the exact behavior that we expected to see during the frequency sweep measurements. Our prediction for the tangent of the phase based on Kirchoff's laws of the real circuit is

$$\tan(\varphi) = -\frac{R+r}{R^2C_X}\frac{1}{\omega} - rC_X\omega$$

So at lower frequencies, we should see a very fast drop-off proportional to ω^{-1} and at high frequencies we should see a linear increase with respect to ω. The first trial, though at very low resolution and with relatively few data points, shows a rapid drop-off at low frequency and increases with increasing frequency. The second trial using a 100pF standard capacitor and 330pF measured capacitor does not show the sharp drop-off for low frequencies, though it shows a relatively linear increase up to about 500kHz before it begins to decrease, which was not expected. The final trial using μF capacitors shows a very clear ω^{-1} for low frequencies, but does not increase at all with higher frequencies. These unexpected behaviors could be due to errors in the programming, lack of precision of measurements, or due to non-linear features of the capacitors and resistors for high-frequencies or high voltage amplitudes. Additionally, when a series inductance L is considered in the real model of the capacitor, the prediction for the tangent of the phase becomes

$$\tan(\varphi) = \frac{-C_S(R+r) - \omega^2 rR^2C_SC_X^2}{\omega(LC_S + R^2C_SC_X) + \omega^3LC_SR^2C_X^2}$$

This means that a high enough inductance can suppress the linear proportionality, so it may have been the case that our model did not take enough into account.

Future work on this system will work to make the measurements of the phase and amplitude more precise, as well as integrate the Python function into a LabVIEW UI once the measurements become more accurate.

References

Capacitance bridge paper

- Lu Li, et al. “Very Large Capacitance Enhancement in a Two-Dimensional Electron System.” Science (American Association for the Advancement of Science), vol. 332, no. 6031, American Association for the Advancement of Science, 2011, pp. 825–28, doi:10.1126/science.1204168.

Quantum Capacitance references

- Serge Luryi (1988). "Quantum capacitance devices." *Applied Physics Letters*. 52 (6): 501–503.
- Datta, Supriyo. “Lecture 30: Quantum Capacitance.” ECE 453: Fundamentals of Nanoelectronics, Purdue University.

OpenDACS resources

- Signal generator: https://opendacs.github.io/projects/acbox/
- DC voltage source: https://opendacs.github.io/projects/dac-adc/
- Arduino code: https://github.com/afylab/Electronics/blob/master/ACBOX_DUAL_AD9854/DUAL_AD9854_due_v3/DUAL_AD9854_due_v3.ino

Root-finding algorithm references

- Newman, Mark. *Computational Physics*. Mark Newman, 2012.

Figures

[1] from http://hyperphysics.phy-astr.gsu.edu/hbase/electric/capac.html

[2] Liu, X., Hao, Z., Khalaf, E. *et al.* Tunable spin-polarized correlated states in twisted double bilayer graphene. *Nature* **583,** 221–225 (2020)

[3] from https://www.physics-and-radio-electronics.com/electronic-devices-and-circuits/semiconductor/intrinsic-semiconductor/fermi-level-in-intrinsic-semiconductor.html

[6] from https://en.wikipedia.org/wiki/Bisection_method

[7] from https://tutorial.math.lamar.edu/classes/calci/newtonsmethod.aspx