
SHAPIRO STEPS IN ARRAYS OF JOSEPHSON JUNCTIONS

February 9, 2022

Student ID:B718539
Loughborough University
Department of Science

Contents

1	Abstract	3
2	Introduction	3
3	Theory	3
3.1	Superconductors	3
3.2	Josephson junction	4
3.3	Josephson Effects	5
3.4	AC Josephson Effect	7
3.5	DC Josephson Effect	7
3.6	RCSJ model	8
3.7	Shapiro Steps	9
3.8	Importance/Previous Studies	9
4	Method	10
4.1	Setup/Data collection	10
4.2	Procedure	11
5	Results	31
6	Discussion	35
7	Conclusion	36
8	Acknowledgments	36
9	Appendix	38
9.1	Diary of meetings	38
9.2	Contribution	38

1 Abstract

An array of 10 parallel Josephson junction which consists of YBaCuO superconductors and a thin film material with a bicrystal grain boundary was tested by an applied magnetic field which produced 4 point measurement which are the current voltage $\frac{dI}{dV}$ and the magnetic field. Matlab was used to create a set of functions that analyse all 600 data files and they convincingly show that there are 4 Shapiro Steps, 2 on the positive side of the magnetic field and 2 on the negative side of the magnetic field, and it highlights that as the order of the Shapiro steps increase the height of the Shapiro steps decay that could be due to the asymmetry of the array.

2 Introduction

The aim of this project is to show the number of Shapiro steps and the height of them from the data which contained the current, voltage, $\frac{dI}{dV}$ and magnetic field that was provided through the testing array of 10 parallel Josephson Junctions. This paper will have a theory where it will go over some of the important background needed such as the Meissner effect in type 2 superconductor, the type of superconductor being worked on (YBaCuO superconducting thin film material with a bicrystal grain boundary) and it will derive the equation for the phase change and the equation for current as a function of phase change, it will also derive the AC and DC Josephson effects, and demonstrate the effectiveness of the RCSJ model in describing the Shapiro steps and the AC and DC Josephson effect, it will also mention previous studies conducted on Josephson junction and Shapiro steps and how they can shape and affect the world as a whole. How the experiment was conducted is shown in the method section, and it also provides a detailed explanation of the function were created to analyse the heights against the magnetic field as well as the difference in height against the magnetic field, the output for these graphs are shown in the results and discussion.

3 Theory

In this section, there is going to be an explanation of superconductors, Josephson junction and their AC and DC Josephson effects and make comparison to the RCSJ model and to wider literature.

3.1 Superconductors

In 1911, Kamerlingh-Onnes designed an experiment [1] which measured the decay time of magnetically induced current in a superconducting ring in which he found that the induced current did not decay as long as the ring is a superconductor, this is because superconductors have little to no resistance. Meissner and Oschensfeld in 1933, investigated this further and found that superconductors repel magnetic fields as mentioned in [2] whereby it states "a normal metal with a magnetic field inside will expel this field when cooled to superconductivity." this is referred to as the Meissner effect, and only works until a critical magnetic field. Both type 1 and type 2 superconductors experience the Meissner effect until their respective critical magnetic field. Since the experiment was done with a type 2 high-temperature superconductor, namely Yttrium barium copper oxide (YBaCuO), they will be just be the topic of discussion. When saying high-temperature this is in-comparison to low temperature superconductors, for

instance high-temperature superconductor has a transitional temperature of about 77K and use liquid nitrogen to cool it down, while the low temperature superconductors use liquid hydrogen to cool down past the transitional temperature of about 33k, from this one can conclude that it is much cheaper to use high-temperature superconductors than low temperature. YBaCuO can operate at a higher temperature relatively speaking so this can be called a type 2 superconductor, not only can the type 2 superconductor experience the meissner effect, but if you exceed the critical magnetic field then you get the Shubnikov phase where magnetic fields partly enter the in the superconductor and so does the shielding current, this causes the magnetic field lines to concentrate such that the flux lines are generated , called Abrokosov vortices [3][4], this occurs above the first critical magnetic field and below the second critical magnetic field, but if the magnetic field exceeds the second magnetic field then the superconductor will just become and ordinary metal. This can be seen in figure 1, the region of interest

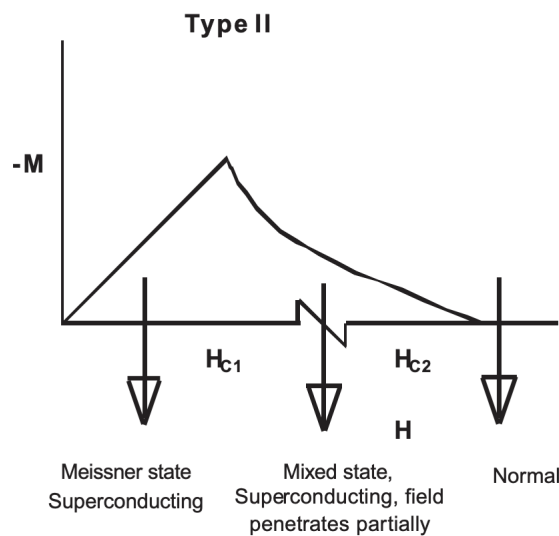


Figure 1: Magnetic vs temperature graph, shows the Meissner state as well as the mixed state which is the Abrokosov vortices [5]

3.2 Josephson junction

Josephson junction contains two superconductors sandwiching an insulating layer. The insulating layer has to be thin enough for electron to tunnel across [6], this is referred to as quantum tunneling [7]. Quantum tunneling allows for microscopic particles to have a finite probability to travel through certain object where as classically this would not be allowed, in this case the electrons crossing an insulating barrier [8]. There are many different type of junction, the type that was used for the experiment was the YBaCuO superconducting thin film material with a bicrystal grain boundary. The experiment was done in the meissner state (Figure 1), this means that the two superconductors will repel the magnetic field, but it can enter the bicrystal grain boundary, and since it was done with a flow of current, this will effect the grain boundary by creating flux flow resonance, this is done by the Lorentz force, as the current travels through the grain boundary the magnetic field will be perpendicular to it, and this will create a force that will move the vortices from one end to the other, this can be seen on the I-V graphs as peaks. The current flowing is linked from one superconductor to the other and is called the Josephson current

[9]. The angle of the grain-boundary can affect the critical current density, for small angles below 20 degrees no Josephson junctions are formed and for angles between 24 to 45 the critical current density strongly decreases monotonically.

3.3 Josephson Effects

Most of the derivation on this section are from the following sources [10][11][12][13][14]. In the previous sections, we talked about the junction and the type of superconductor we are using, here we are going to derive some important equation. Although the experiment conducted uses magnetic fields, but to derive the equation for the Josephson junctions we can consider a simpler scenario whereby there is no magnetic field and the junction is symmetrical. Since the two superconductors are separate systems therefore they are governed by two different wave function which according to the time-dependent Schrödinger equation is,

$$i\hbar \frac{\partial \Psi_1}{\partial t} = E_1 \Psi_1; \quad i\hbar \frac{\partial \Psi_2}{\partial t} = E_2 \Psi_2 \quad (1)$$

Ψ_1 is the amplitude to find the electron in the first superconductor and Ψ_2 is the amplitude to find the electron in the second superconductor, and E_1 and E_2 is the energy of each superconductor (REF 10). Since the electrons are tunneling from one superconductor to the other, then there is a weak coupling strength between the superconductors and this coupling strength is fixed by a constant G . If you connect a voltage between the two superconductors it will be governed by $E = qU$ where $E = E_1 - E_2$ and q is the charge and U is the voltage, if you take the zero point energy to be half way, we can write this as $E = \frac{qU}{2}$, so this will make equation 1 become the following

$$i\hbar \frac{\partial \Psi_1}{\partial t} = \left(\frac{qU}{2} \Psi_1 + G \Psi_2 \right) \quad (2)$$

$$i\hbar \frac{\partial \Psi_2}{\partial t} = \left(-\frac{qU}{2} \Psi_2 + G \Psi_1 \right) \quad (3)$$

We can make a guess (ansatz) for the wave-functions which is the following:

$$\Psi_1 = \sqrt{p_1} e^{i\theta_1}; \quad \Psi_2 = \sqrt{p_2} e^{i\theta_2} \quad (4)$$

p_1 and p_2 are the density of the electrons and θ_1 and θ_2 is the phase of the wave-functions Ψ_1 and Ψ_2 respectively (i.e Phase between the two superconductors). We can then substitute equation 4 into equation 2 and 3, we get the following equations;

$$i\hbar \frac{\partial}{\partial t} \left(\sqrt{p_1} e^{i\theta_1} \right) = \left(\frac{qU}{2} \sqrt{p_1} e^{i\theta_1} + G \sqrt{p_2} e^{i\theta_2} \right) \quad (5)$$

$$i\hbar \frac{\partial}{\partial t} \left(\sqrt{p_2} e^{i\theta_2} \right) = \left(-\frac{qU}{2} \sqrt{p_2} e^{i\theta_2} + G \sqrt{p_1} e^{i\theta_1} \right) \quad (6)$$

For both LHS of equation 5 and 6 we can take implicit derivatives with respect to the density and the phase while keeping the RHS the same, this will be the following:

$$i\hbar \left(\frac{1}{2\sqrt{p_1}} e^{i\theta_1} \frac{\partial p_1}{\partial t} + i\sqrt{p_1} e^{i\theta_1} \frac{\partial \theta_1}{\partial t} \right) = \left(\frac{qU}{2} \sqrt{p_1} e^{i\theta_1} + G \sqrt{p_2} e^{i\theta_2} \right) \quad (7)$$

$$i\hbar\left(\frac{1}{2\sqrt{p_2}}e^{i\theta_2}\frac{\partial p_2}{\partial t} + i\sqrt{p_2}e^{i\theta_2}\frac{\partial \theta_2}{\partial t}\right) = \left(-\frac{qU}{2}\sqrt{p_2}e^{i\theta_2} + G\sqrt{p_1}e^{i\theta_1}\right) \quad (8)$$

If we now focus on equation 7 and we times both sides of the equation with $\sqrt{p_1}e^{-i\theta_1}$, also to make life simple we can write $\frac{\partial p_{1,2}}{\partial t} = \dot{p}_{1,2}$ and $\frac{\partial \theta_{1,2}}{\partial t} = \dot{\theta}_{1,2}$ as a short-hand expression, and the following result will be:

$$\left(\frac{i\hbar\dot{p}_1}{2} - \hbar p_1\dot{\theta}_1\right) = \frac{qU}{2}p_1 + G\sqrt{p_1p_2}e^{i(\theta_2-\theta_1)} \quad (9)$$

Since $e^{i\theta} = \cos(\theta) + i\sin(\theta)$, we can compare the imaginary coefficient and the real coefficient, for the density we compare the imaginary parts and since we have $e^{i(\theta_2-\theta_1)}$ on the LHS we can use the identity to the imaginary part which is $e^{i(\theta_2-\theta_1)} = i\sin(\theta_2-\theta_1)$ we can put this back into the equation and we find that we get

$$\frac{\hbar\dot{p}_1}{2} = G\sqrt{p_1p_2}\sin(\theta_2-\theta_1) \quad (10)$$

if we now solve for \dot{p}_1 , and make $\theta_2-\theta_1 = \delta$ we get the following result

$$\dot{p}_1 = \frac{2}{\hbar}G\sqrt{p_1p_2}\sin(\delta) \quad (11)$$

If we go back to equating 9 and now compare the coefficient to all the real parts we, since on the LHS there is $e^{i(\theta_2-\theta_1)}$ we can use the real part of that which is $e^{i(\theta_2-\theta_1)} = \cos(\theta_2-\theta_1)$ using $\theta_2-\theta_1 = \delta$ we get the following result:

$$-\hbar p_1\dot{\theta}_1 = \frac{qU}{2}p_1 + G\sqrt{p_1p_2}\cos(\delta) \quad (12)$$

If we now solve for $\dot{\theta}_1$ we get the following result:

$$\dot{\theta}_1 = -\frac{G}{\hbar}\sqrt{\frac{p_2}{p_1}}\cos(\delta) - \frac{qU}{2\hbar} \quad (13)$$

Since this is only done for equation 7, we can do the same process for equation 8, by substituting $\sqrt{p_2}e^{-i\theta_2}$ and comparing coefficients just like equation 7, we get the following result for the density and the phase (when i say density and phase, i mean that partial derivative with respect to time):

$$\dot{p}_2 = -\frac{2}{\hbar}G\sqrt{p_1p_2}\sin(\delta) \quad (14)$$

$$\dot{\theta}_2 = -\frac{G}{\hbar}\sqrt{\frac{p_1}{p_2}}\cos(\delta) + \frac{qU}{2\hbar} \quad (15)$$

So equations 11,13,14 and 15 is what we get when we compare the coefficients for the imaginary and real parts from equations 7 and 8. From equation 11 and 14 we can see that $\dot{p}_1 = -\dot{p}_2$, this tells us how the current starts to flow from the first superconductor to the second superconductor this could be \dot{p}_1 or $-\dot{p}_2$, but if we assume that the two superconductors are the same $p_1 = p_2$ we get the following result:

$$I = \frac{2Gp_1}{\hbar}\sin(\delta) \quad (16)$$

I is the current, and if we substitute $I_c = \frac{2Gp_1}{\hbar}$ into equation we get the following result:

$$I = I_c \sin(\delta) \quad (17)$$

I_c is a number in which is a characteristic of a particular junction. Equation 17 shows that the current that flows through the superconductor is a function of the change in phase. If we now compute the difference in the phase $\delta = \theta_2 - \theta_1$ using equation 13 and 15 and considering the fact that when electrons tunnel through the insulator they go in pairs (cooper pairs) so we have $2q$ rather than q we get the following result

$$-\frac{G}{\hbar} \sqrt{\frac{p_1}{p_2}} \cos(\delta) + \frac{2qU}{2\hbar} - \left(-\frac{G}{\hbar} \sqrt{\frac{p_2}{p_1}} \cos(\delta) - \frac{2qU}{2\hbar} \right) = \frac{2qU}{\hbar} \quad (18)$$

So the two important equations are the following:

$$I = I_c \sin(\delta) \quad (19)$$

$$\frac{d\delta}{dt} = \frac{2qU}{\hbar} \quad (20)$$

We can write equation 20 in another way, noticing that $\hbar = \frac{h}{2\pi}$ and that the magnetic flux through a loop and is quantized and is called the magnetic flux quantum which is $\Phi_0 = \frac{h}{2q}$, if we replace this into equation 20 we get the following result;

$$\frac{d\delta}{dt} = \frac{2\pi U}{\Phi_0} \quad (21)$$

From these equations we can get the AC and DC Josephson effects

3.4 AC Josephson Effect

If we have a constant DC voltage that drives the junction, where $V = \text{const}$, if we directly integrate both sides of equation 20 by dt which is for the phase change we will then get the following result[15]

$$\delta = \frac{2qUt}{\hbar} \quad (22)$$

Where the $\delta = \theta_2 - \theta_1$ is the change in phase, if we then plug equation 22 into equation 19, we get the following result[15]

$$I(t) = I_c \sin\left(\frac{2qUt}{\hbar}\right) \quad (23)$$

With the AC Josephson Effect we can see that the DC voltage that is driving the current, causes the current to oscillate as a function of time. So a DC voltage causes an AC current, this is why the junction can act as a perfect voltage to frequency converter[15].

3.5 DC Josephson Effect

Since keeping the voltage across the junction is not always possible, the best thing to do is to keep the current constant, so $I(t) < I_c$, for this to happen the $\sin(\delta)$ can not change, so the δ is therefore constant as well, and from equation 20 this would mean that $\dot{\delta} = 0$, therefore the $V = 0$. So the change in phase

will equal the following from equation 19[15]

$$\delta = \sin^{-1} \left(\frac{I}{I_c} \right) \quad (24)$$

The phase is constant, but the change in phase is given by equation 19. What this means is that in the absence of voltage constant super-current can still flow through the insulating barrier [16][15].

3.6 RCSJ model

The Resistance capacitor shunted junction (RCSJ) model can describe the current-voltage characteristics but also the dynamics of Josephson junctions [17]. The RCSJ model consists of a resistor, capacitor, Josephson junction in parallel, with a potential difference, this can be seen in figure 2

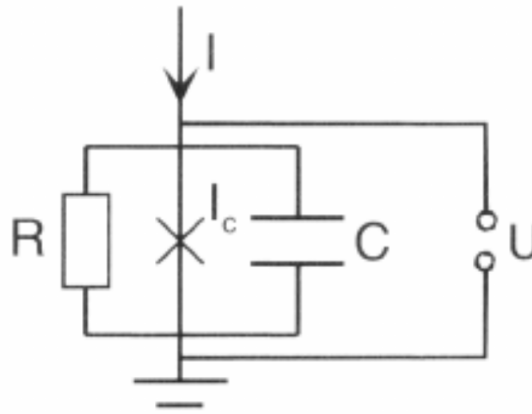


Figure 2: The RCSJ model [17]

We can use Kirchhoff's current law, whereby it states that the total current enclosed by a path is the algebraic sum of the individual components, in this case the equation is[17];

$$I_t = I_j + I_r + I_c \quad (25)$$

Where I_t is the total current, I_j is the Josephson current, I_r is the current through the resistor and I_c is the current through the capacitor can be thought as the displacement current. We can use equation 19 for I_j , $I_r = \frac{U}{R}$ and $I_c = C\dot{U}$, this will then be[17]:

$$I_t = I_c \sin(\delta) + \frac{U}{R} + C\dot{U} \quad (26)$$

We can use equation 21 to get all of these in terms of the phase and the flux quantum, by rearranging for U and then plugging it into equation 26, we get the following result [17]

$$I_t = I_c \sin(\delta) + \frac{\Phi_0}{2\pi R} \dot{\delta} + \frac{C\Phi_0}{2\pi} \ddot{\delta} \quad (27)$$

This is analogous to a second order mechanical systems which is the pendulum and its equations is [17]

$$M = mgl \sin(\gamma) + \Gamma \dot{\gamma} + \Theta \ddot{\gamma} \quad (28)$$

where Γ is the damping coefficient of the pendulum and Θ is the moment of inertia and γ is the deflection angle of pendulum, from this we can make comparison from equation 28 to equation 27, we can see that $I_c \rightarrow mgl$, $\Gamma \rightarrow \frac{\Phi_0}{2\pi R}$ and $\Theta \rightarrow \frac{C\Phi_0}{2\pi}$. We can just look at the motion of the pendulum instead of looking at the Josephson junction, so for small deflection of the angle the average time velocity $\langle \dot{\gamma} \rangle = 0$, for the Josephson junction this means that the average time voltage $\langle U \rangle = 0$, this is similar to the DC Josephson effect. If the deflection angle is 90° for the junction this would mean that $I = I_c$ and each further increase in torque will rotate the pendulum, but if the damping is large the motion becomes very nonuniform, but for larger and larger torque the average time velocity $\langle \dot{\gamma} \rangle$ will become proportional to M , for the Junction this would mean that the average time voltage $\langle U \rangle$ will become proportional to $I(\text{Current})$ [17]. We can use this to generate a Current-voltage characteristic which can be seen in the Results and conclusion.

3.7 Shapiro Steps

With the necessary perquisites done we can discuss the Shapiro Steps, if continue with the RSCJ model analogy but now we apply an alternating current, this can be seen as a driven pendulum and at certain interval of torque values M , the pendulum will rotate with exactly the driving frequency, this means that there is range where $\langle \dot{\gamma} \rangle$ is constant, if we apply this to the junction this means that there is a certain range where the $\langle U \rangle$ is constant. If we then make $\delta = 2\pi f_{ac} t$, where f_{ac} is the alternating frequency and t is the time and replace it in equation 21 and the U is the $\langle U \rangle$ we get the following [18]

$$\frac{d}{dt}(2\pi f_{ac} t) = \frac{2\pi \langle U \rangle}{\Phi_0} \quad (29)$$

We then take the derivative and cancel the 2π out and solve for the $\langle U \rangle$, we get the following result [18]

$$\langle U \rangle = n f_{ac} \Phi_0 \quad (30)$$

Equation 30 shows us that at integer multiple values of the frequency we get constant voltage, this can be seen clearly if you take derivatives of the current-voltage values for the specific Josephson Junctions data [18]

3.8 Importance/Previous Studies

In the previous parts we have discussed some of the features of superconductors as well as Josephson Junction and its effects. We can start by talking about, the grain-boundary of our Josephson junction its importance is highlighted by [19] whereby it states "research on bi-crystals is crucial for high-temperature superconductor development", and as mentioned in section 3.1 high-temperature superconductors are cheaper this makes them more readily available to use rather than having to employ the use of liquid nitrogen, the overall aim to have a room-temperature superconductor and studying the bi-crystals can develop an understanding of how to achieve that, with that in mind the Josephson Junction can be used

to reduce noise in a quantum processor so it can develop cat states [20][21], not only that they can be used in superconducting quantum interference device (SQUIDS) which are sensitive to magnetic fields and can detect subtle changes, SQUIDS can be used in the medical field for heart and brain current, and can be used by geologist to investigate slight magnetic field changes on earth [22][23]. Although what is presented here is limited but it shows that these junctions are a fundamental aspect of any applications, so understanding what phenomena happen in these junctions are important such as Shapiro Steps, flux-flow resonance and Fiske steps. We are just focusing on the Shapiro steps and they have only been observed up to 2.5THz [24], but they theoretically can be observable up to 10-20THz frequency range for high-temperature superconductors, the reason as to why they have been observed up to 2.5THz is because they are limited by the energy gap of the superconductors as suggested by [18] "Microscopic theory indicates that I_c strongly decreases for frequency above $\frac{\Delta_0}{h}$, microwave frequency above this value, the amplitude of the Shapiro steps rapidly approaches zero" where Δ_0 is the energy gap in the superconductor, getting the maximum frequency up to which Shapiro steps can be observed is desired, a study in which uses a Superconducting Single Electron Transistor irradiated with microwaves up to 18 GHz observe the height in current and the step position in voltage of the Shapiro Steps in the positive DC IV characteristics, ignoring the voltage at 0 as this is caused due to the DC Josephson effect they find two in positive part with different heights [25]. There also has been a study that observes Shapiro steps in the absence of microwave radiation [26], they use a larger voltage, and observe 4 peaks in the $\frac{dI}{dV}$ vs Voltage graph which are the Shapiro steps, they ignore the one at the 0 peak as that is the DC Josephson effect, also study has shown Shapiro steps in alternating current densities [9].

4 Method

4.1 Setup/Data collection

The experiments conducted consisted of 10 Josephson junction in parallel made of YBaCuO film material with a bicrystal grain boundary. The array was asymmetric, this means that the area of the junctions increase as you moved further down as shown in figure 3

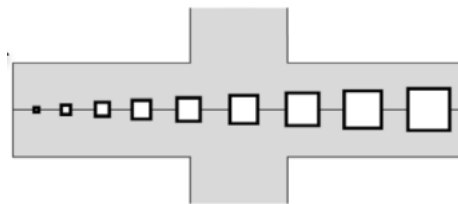


Figure 3: asymmetric array of Josephson Junctions(REF 22)

600 data files were produced and in each data consists of the dc current, dc voltage and $\frac{dI}{dV}$ as well as the applied magnetic field. The magnetic field is varied from -3.1mT to +2.9mT.

4.2 Procedure

Data was collected for me, to then analyse and produce graphs that shows the height (as in max data point subtracted from the min data point) of the peaks with the applied magnetic field for each 600 data files, this process was done on matlab. Creating a set of function is a script file in matlab that can compute any sort of data presented and provide reasonable precise and accurate measurements for the height of Shapiro steps. Initially a function was made to import all the data into matlab, this was done via the code below:

```

1  sDirPath='/Users/mohammedusmaan/MWPower/';
2
3  function FilesData = ReadDirectry(pDirPath)    % Function remains at
    bottom
4      iRowCnt=0;
5      sAllFiles='*.*';
6      lstFiles=dir(sprintf('%s%s',pDirPath,sAllFiles));
7      szFiles=size(lstFiles);
8      colFileData= cell(szFiles-2);
9      for iCnt=1:szFiles
10         if iCnt>2
11             iRowCnt=iRowCnt+1;
12             sFileNames=lstFiles(iCnt).name;
13             sFilePath=lstFiles(iCnt).folder;
14             sFilepath_Filename=sprintf('%s/%s',sFilePath,sFileNames);
15             colFileData{iRowCnt} = load(sFilepath_Filename);
16         end
17     end
18     FilesData = colFileData;
19
20 end

```

Listing 1: Imputing all the 600 data into the work space of matlab

The data files are numbered extension (i.e filename.07) not txt extensions (i.e filename.txt), so imputing all the files is a bit more challenging, with that in mind the first two lines of the 1st listing is put at the top of the script while the function was put at the bottom of the script file otherwise the data does not get imputed. The function requires the directory (i.e path) to where the 600 files are located, the 1st line of the code does that, you then pass that thought the function, line 6 uses this directory in conjunction with the (.) specified in line 5 to list all the files in the folder this can be seen in figure 4

Fields	name	folder	date	bytes	isdir	datenum
1	'.'	'/Users/...	'03-Feb-...	0	1	7.3782e+05
2	'..'	'/Users/...	'14-May-...	0	1	7.3793e+05
3	'2607201...	'/Users/...	'02-Feb-...	37599	0	7.3782e+05
4	'2607201...	'/Users/...	'26-Jul-2...	37601	0	7.3617e+05
5	'2607201...	'/Users/...	'26-Jul-2...	37600	0	7.3617e+05
6	'2607201...	'/Users/...	'26-Jul-2...	37601	0	7.3617e+05
7	'2607201...	'/Users/...	'26-Jul-2...	37601	0	7.3617e+05
8	'2607201...	'/Users/...	'26-Jul-2...	37602	0	7.3617e+05
9	'2607201...	'/Users/...	'26-Jul-2...	37601	0	7.3617e+05
10	'2607201...	'/Users/...	'26-Jul-2...	37601	0	7.3617e+05
11	'2607201...	'/Users/...	'26-Jul-2...	37602	0	7.3617e+05
12	'2607201...	'/Users/...	'26-Jul-2...	37603	0	7.3617e+05
13	'2607201...	'/Users/...	'26-Jul-2...	37602	0	7.3617e+05

Figure 4: The output of line 6

This only shows the name of the files but not content in the file which is are the dc current, dc voltage, $\frac{dI}{dV}$ and the applied magnetic field. line 8 removes the first two names as they are not contain no bytes of information and creates empty cells for all the file to be located in. line 9 to 17 is the for loop, this is then going down the list and reading the content of each file in the folder and placing them into empty cell that was creating before the loop, line 18 is the output of all the files, this can be seen in figure 5

	1	2	3	4	5	6	7	8	9	10	11	12	13
1	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...	800x4 do...

Figure 5: Filled cells of all the data files

If you open one of these files, it will show the current, voltage, $\frac{dI}{dV}$ and the applied magnetic field. To illustrate this the first data files measurements can be seen in figure 6, where the 1st column is the voltage, the 2nd column is the current, the 3rd column is the $\frac{dI}{dV}$ and the 4th column is the B field value.

colFileData{1, 1}				
	1	2	3	4
1	7.2766e-...	-8.0000e...	-0.3872	0.0031
2	1.5674e-...	-7.9800e...	363.0480	0.0031
3	-7.2869e...	-7.9600e...	365.1260	0.0031
4	-7.2655e...	-7.9400e...	-1.0684	0.0031
5	-7.2466e...	-7.9200e...	-0.9461	0.0031
6	-7.2263e...	-7.9000e...	-1.0153	0.0031
7	-7.2077e...	-7.8800e...	-0.9300	0.0031
8	-7.1884e...	-7.8600e...	-0.9623	0.0031
9	-7.1694e...	-7.8400e...	-0.9530	0.0031
10	-7.1496e...	-7.8200e...	-0.9853	0.0031
11	-7.1307e...	-7.8000e...	-0.9484	0.0031
12	-7.1113e...	-7.7800e...	-0.9692	0.0031
13	-7.0920e...	-7.7600e...	-0.9646	0.0031
14	-7.0728e...	-7.7400e...	-0.9600	0.0031
15	-7.0525e...	-7.7200e...	-1.0130	0.0031
16	-7.0325e...	-7.7000e...	-1.0015	0.0031
17	-7.0131e...	-7.6800e...	-0.9715	0.0031
18	-6.9938e...	-7.6600e...	-0.9646	0.0031
19	-6.9739e...	-7.6400e...	-0.9969	0.0031
20	-6.9543e...	-7.6200e...	-0.9784	0.0031

Figure 6: All the content of the 1st data file

Now the all 600 files are uploaded in matlab, we can now separate each of the measurements by creating cells for the current, voltage, $\frac{dI}{dV}$ and the magnetic field, and can use the cells to plot all the calculated $\frac{dI}{dV}$ against voltage and compare to the measured $\frac{dI}{dV}$ against voltage this is done by the following code:

```

1 colFilesData = ReadDirectory(sDirPath);
2 for iColFileCnt = 1:length(colFilesData)
3
4     EveryFile = colFilesData{1,iColFileCnt}
5     [RowEveryfile, ColEveryfile] = size(EveryFile)
6     AllB_field(iColFileCnt,1) = EveryFile(1,4)
7     AllVoltage{1,iColFileCnt} = EveryFile(:,1)
8     AllCurrent{1,iColFileCnt} = EveryFile(:,2)
9     AllDIDV{1,iColFileCnt} = EveryFile(:,3)
10
11 end
12
13 for i = 1:length(colFilesData)
14     CalDIDV = gradient(AllCurrent{1,i})./gradient(AllVoltage{1,i})
15     SCalDIDV = smooth(CalDIDV,10)
16     plot(AllVoltage{1,i},SCalDIDV,'k-')
17     hold on
18     plot(AllVoltage{1,i},AllDIDV{1,i},'b-')
19     xlabel('Voltage')
20     ylabel('dI/dV (1/\Omega)')
21     ylim([-4 4])

```

```

22 title('Calculated dI/dV and Measured dI/dV against Voltage')
23 end

```

Listing 2: Code to sort all the data and to plot

In code listing 2, line 1 is the output data from the 1st listing function and it is used in the for loop from line 2 to 10 which goes through each of the 600 files and separates them into their individual cells just like with figure 5, where now if you click on lets say the AllVoltage variable which is on line 7 it will show 800x1 for each cell 600 times, this is the same for all the files except for the magnetic fields as we just need to extract it once per file not 800 times. We can use individual cells in particular the voltage and the current to calculate $\frac{dI}{dV}$ and plot against the measured $\frac{dI}{dV}$, this is done from line 13 to line 23, the for loop will cycle through all the relevant individual cells, line 14 calculates $\frac{dI}{dV}$ from the current and the voltage, line 15 smooths the graph out by taking the mean every 10 values, lines 16 and 18 plots the data on the same graph and line 21 limits the y values from -4 to 4 so we can see the plots. The result can be seen in figure 7

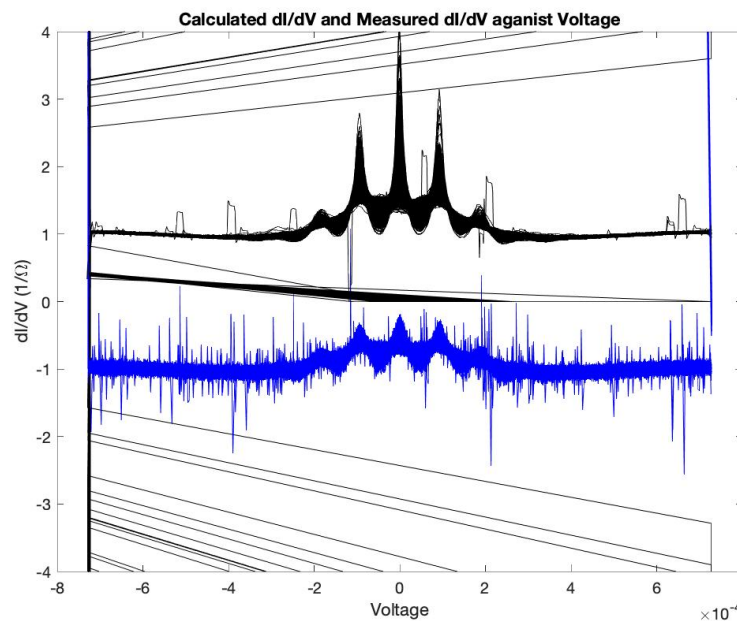


Figure 7: Black is the Calculated $\frac{dI}{dV}$ and Blue is the measured $\frac{dI}{dV}$

This figure is used as an indicator to show how many peaks there are, and where in the y-axis they are located, this information is needed so we can remove any unwanted peaks that don't provide us useful information regarding the Shapiro steps. Although from the figure there are a lot of spikes in the data we can make out 4 distinct peaks 2 on the negative voltage and 2 on the positive voltage, we can say that there are two Shapiro steps on the negative voltage and 2 Shapiro Steps on the positive voltage, we can visually see the Shapiro steps now we need to highlight this in the code using the function called "findpeaks", this requires the y-value of the smallest peaks on each side which is about 1.09, but there are some issues using this function as it can highlight data points that are not the peaks as illustrated in figure 8.

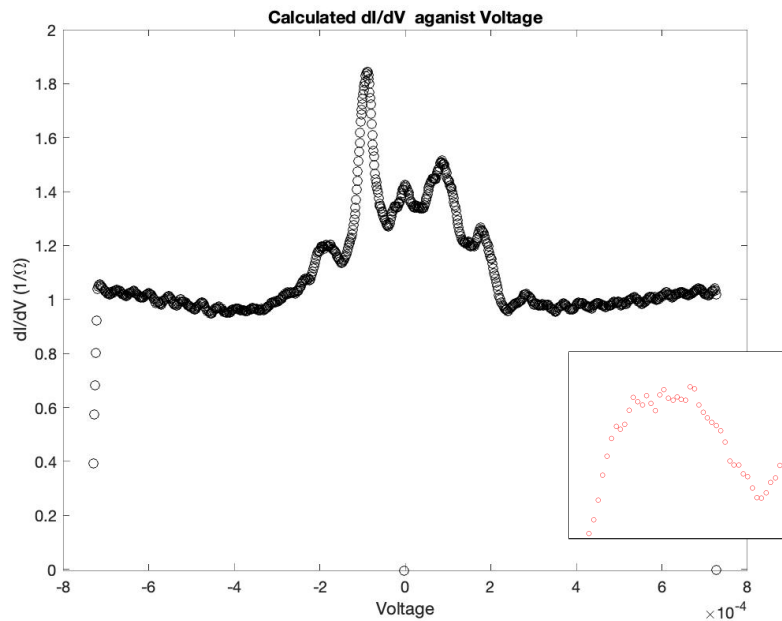


Figure 8: 4th data file for $\frac{dI}{dV}$ against the voltage, showing a zoomed in view of the 2nd negative peak location is x-value = -2×10^{-4}

Figure 8, in the zoomed window you can see many points close together and the findpeaks function can't distinguish between which is the actual max point and which is not, also we rotated the graph by 180° and used the findpeaks function to find them min values, the same issue occurs for the min values. The findpeaks will also pick out the zeroth voltage min and max value, this is not a Shapiro steps but the DC Josephson effect, so in calculating the height we can't consider this. To calculate the height consider figure 9.

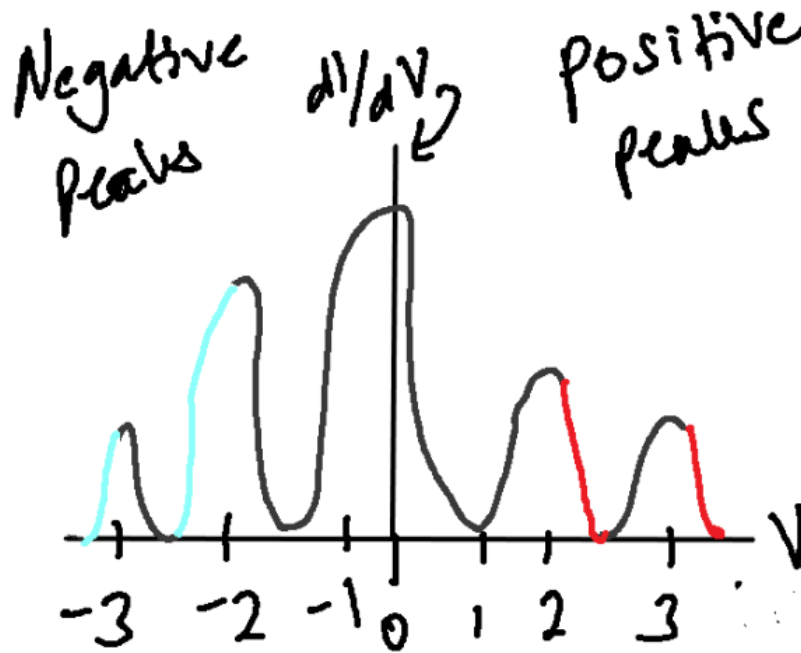


Figure 9: Showing the Shapiro steps, highlighted parts show for calculating height the voltage is in $\times 10^{-4}$

The blue lines indicate which sides we are taking the Minima(min) and Maxima(max) points from, the same goes for the red lines. We then filter the min points out, so as to take the sides we need also we need to remove the min and max points on either side of 0. We can start by using the findpeak function to find the max points this is done by the code below.

```

1  %%%% Initialize Peak Length Data %%%%%%%%%%
2  iPeakLens=zeros(iColFileCnt,4);
3  %Max Peak Height
4  iMinPksHgt=1.09;%1.215;%1.3;
5  %Max Peak Distance
6  iMinPksDist=35;%35;
7  %Max Peaks Prominence
8  iMinPksProm= 0.00515;%0.01915;%0.01;%0.00515;%
9
10 function [maxPks,maxLocs,maxWidth,MaxProm,sCalDiDvData]=
    EvaluateDataForPeaks(pVoltRefs,pCurrentRefs,pMPH,pMPD,pOutputPlot)
11
12 CalDIDV = gradient(pCurrentRefs)./gradient(pVoltRefs);
13 SCalDIDV = smooth(CalDIDV,10);
14 [pks,locs,w,p] = findpeaks(SCalDIDV,'MinPeakHeight',pMPH,'
    MinPeakDistance',pMPD); % Anything below 1.09 gets cut off.
15 if pOutputPlot==1

```



```

16         plot(pVoltRefs,SCalDIDV,'ro');
17         hold on;
18         plot(pVoltRefs(locs),pks,'co');
19     end
20     maxPks=pks;
21     maxLocs=locs;
22     maxWidth=w;
23     MaxProm=p;
24     sCalDiDvData=SCalDIDV;
25
26 end

```

Listing 3: Code to find the max and the min points for all the graphs

In listing 3, from line 2 to 8 as the code states we are initializing values to input into the findpeaks function, these values are based on figure 7 plot, these then get passed through the function on line 10 with the current and voltage, line 12 and 13 are the same from listing 2, line 14 evaluates the max points for all $\frac{dI}{dV}$ against voltage, this is done by inputting the threshold for the lowest max point on the curve which is 1.09, and the distance between two points which is 35, this means that any max point above the y-value of 1.09 is considered a peak and the distance between two peaks is limited to 35 data points as to not have multiple peaks together, and the output from the function is those peaks locations and line 15 to 18 is there so you can plot certain graphs if need be. Figure 10, shows the max peaks found for the first set of data

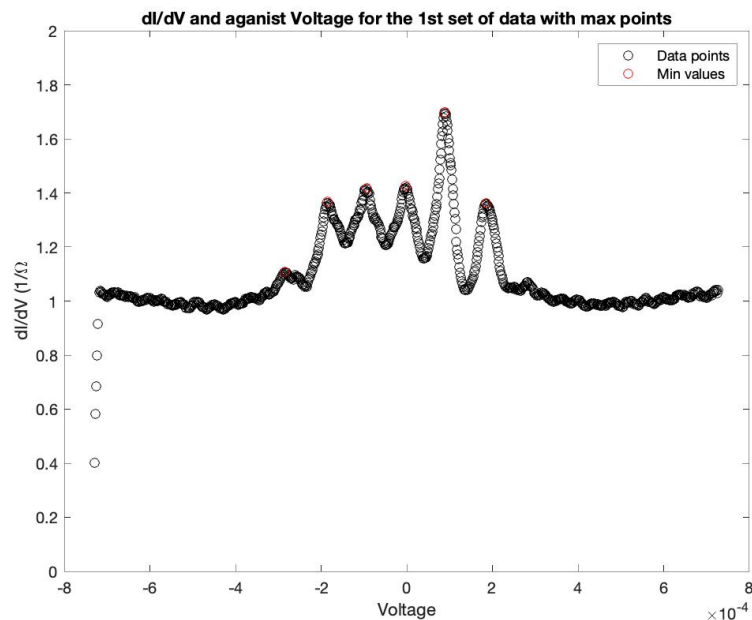


Figure 10: The red dots are the peaks on the graph

From figure 10 we can see that there is a peak found on the 0th voltage, as mentioned before, this needs to be removed as it does not factor into any of our calculations, to do this we need to create a function that checks the x-values of the peaks found and if they are at close proximity to 0, they will be discarded. The code that does this is below

```

1  function [PntsB4Zero,PntsA4Zero]=EvaluateMaxPeaksPostions(pMaxPksData,
    pMaxLocs,pVoltsRef)
2
3      iSizeMaxPks=length(pMaxPksData);
4      iPntb4zero=zeros(iSizeMaxPks,2);
5      iPntA4zero=zeros(iSizeMaxPks,2);
6      iTmpX=0;
7      iZeroPnt=0.0000;
8      iB4Cnt=1;
9      iA4Cnt=1;
10     for iSrchPnt=1:iSizeMaxPks
11         iTmpX=pVoltsRef(pMaxLocs(iSrchPnt));
12         iRndVal=round(iTmpX,4);
13         if iRndVal== iZeroPnt
14             %do nothing;
15         elseif iTmpX>iZeroPnt
16             iPntA4zero(iA4Cnt,1)=iTmpX;
17             iPntA4zero(iA4Cnt,2)=iSrchPnt;
18             iA4Cnt=iA4Cnt+1;
19         else
20             iPntb4zero(iB4Cnt,1)=iTmpX;
21             iPntb4zero(iB4Cnt,2)=iSrchPnt;
22             iB4Cnt=iB4Cnt+1;
23         end
24     end
25     PntsB4Zero=iPntb4zero;
26     PntsA4Zero=iPntA4zero;
27 end

```

Listing 4: Code to disregard the 0th voltage peak

The output from listing 3 code is used as an input for this function, the inputs are the max values, location of max values and the voltage, line 5 is the length of the max data points found, line 6 and 7 are setting up the output for this function, line 8 to 11 is setting up the values for the for loop, and line 12 to line 26 is the for loop. Line 13 in the for loop gets the x-value for the max points found, line 14 rounds the x-value up to 4 decimal places and line 15 is an if statement whereby if the x-value is on zero it does nothing effectively skipping the corresponding peak, line 17 to line 25 then splits the x-value into the variable points after zero and points before zero and those are the output for this function. So far we have found the max points and excluded the max points found at the 0th voltage, now we need to

find the min values, we do this by rotating the graph by 180° and use the findpeaks function, this can be seen in figure 11.

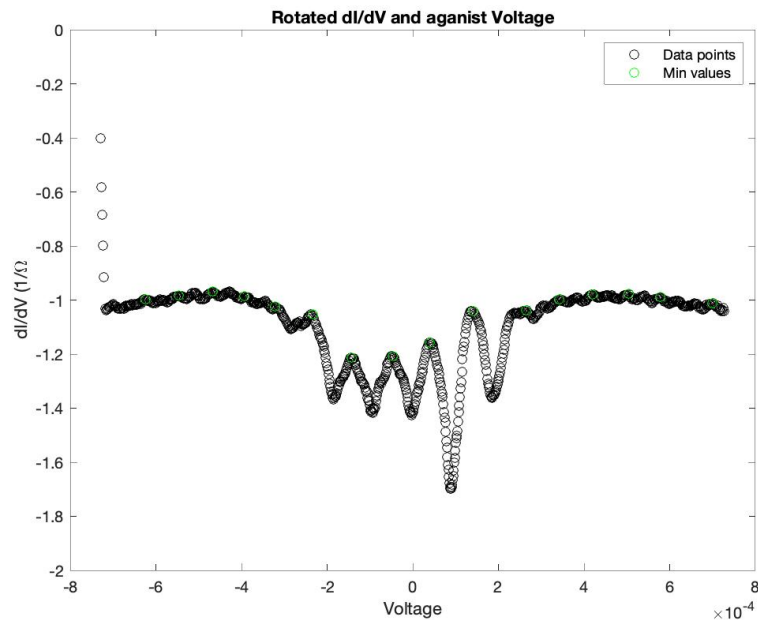


Figure 11: Rotated graph for the first 1st data points with the green dots showing min values from the findpeak function

From figure 11, we can see that the findpeak function collects extra peaks that we do not need, to remove this we need a function that goes through all the min points and remove the ones that don't correspond to figure 9, since the next two functions are the main part of the code it will require more detail as to what is going on.

```

1 function minPksData=EvaluateMinPeaks(pMaxPksData,pMaxLocs,pCalDifData,pMPP,
   pVoltsRef,pDoMinPointEval,pOutputPlot)
2
3 % Point zone area from max peak points
4 % to determine what min points to remove.
5 iPntZoneFrmMax=15;
6
7 %Close point to max peaks tolerance
8 iClosePointTol=0.04;
9
10 %Min peaks close tolerance
11 iMinPksCloseTol=0.065;
12
13 % determining the number of max peak points
14 iSizePks=length(pMaxPksData);
15 iSizeLocs=length(pMaxLocs);

```

```
16 % Number of min points required
17 iMinPntRq=iSizePks+1;
18
19 % Rotate the max points
20 iRotDatapnts=-pCalDifData
21
22 % Find the min peaks with tolerance 01915 - 0.03115 -- This may require
    further
23 % tweeking on overall files. These tolerance produce min peak point
    from 11 to 7
24 % points
25 [minPks,minLocs,minW,minP] = findpeaks(iRotDatapnts,'MinPeakProminence',
    ,pMPP);
26
27 % Rotate the new max points ==> minPks
28 rotMinPksData=minPks;
29 iRotMinPks=-minPks;
30 minPeaksPnts= iRotMinPks;
31 minPeakLoc=minLocs;
32 % determining the number of min peak points
33 iSizeMinPks=length(iRotMinPks);
34 iSizeMinLocs=length(minLocs);
35
36 % Nnumber of elements to be removed
37 iElemToRemove=0;
38
39 %Close Tolerance element to remove
40 iClsTolElm=1;
41
42 % iRemove Element flag
43 iRemoveElemFlag=0;
44
45 % To Min peak elements processed
46 iMinElemProcessed=0;
47
48 % storing positions of the reduntant min peaks
49 iMinPeaksRemoved=zeros(iSizeMinPks,1);
50
51 iNxtStrtCnt=0;
52 %remove closet points to max peaks
53
54 iMinLocChksVal=4;
```

```
55
56 % check number of new max points evaluated by findpeak function
57 % are they greater than min points required.
58 try
59 if iSizeMinPks>iMinPntRq
60
61 %remove closet points to max peaks
62 for iClstPnt=1:iMinPntRq
63     if iClsTolElm>0
64         if iClstPnt<=iSizeLocs
65             iTmpMaxPk=pMaxLocs(iClstPnt);
66             for iChkMinPnt=1:iSizeMinLocs
67                 if iChkMinPnt>iMinElemProcessed
68                     iTmpMinPk=minLocs(iChkMinPnt);
69                     if iTmpMinPk>iTmpMaxPk
70
71                         for iMinLocVal=1:iMinLocChksVal
72                             iNewLocChk=iChkMinPnt-iMinLocVal;
73                             if iNewLocChk>iMinElemProcessed
74                                 iNwTmpMinPk=minLocs(iNewLocChk);
75                                 iNewVldPnt=iTmpMaxPk-iNwTmpMinPk;
76                                 if iNewVldPnt>=iPntZoneFrmMax
77                                     iMinPeaksRemoved(iNewLocChk)=0;
78                                     break;
79                                 end
80                             else
81                                 iMinPeaksRemoved(iChkMinPnt-1)=0;
82                                 break
83                             end
84                         end
85                     end
86
87                     iMinElemProcessed=iMinElemProcessed+1;
88
89                     iRemoveElemFlag=1;
90                     break;
91                 else
92                     iMinPeaksRemoved(iChkMinPnt)=iChkMinPnt;
93                     iRemoveElemFlag=1;
94                 end
95             end
96         end
97     end
98 end
```

```
97         else
98             for iChkMinPnt=1:iSizeMinLocs
99                 if iChkMinPnt>iMinPntRq
100                     iMinPeaksRemoved(iChkMinPnt)=iChkMinPnt;
101                     iRemoveElemFlag=1;
102                 end
103             end
104         end
105         % re-evaluate the number of elements
106         if iRemoveElemFlag>0
107             % Entry point flag for removal min peak point
108             iMinPkEntryFlag=0;
109             for iRemovePnt=1:length(iMinPeaksRemoved)
110                 iTmpPntRm=iMinPeaksRemoved(iRemovePnt);
111                 if iTmpPntRm>0
112                     if iMinPkEntryFlag ==0
113                         iRotMinPks(iTmpPntRm) = [];
114                         minLocs(iTmpPntRm) = [];
115                         iMinPkEntryFlag=1;
116                     else
117                         iRotMinPks(iTmpPntRm-iMinPkEntryFlag) = [];
118                         minLocs(iTmpPntRm-iMinPkEntryFlag) = [];
119                         iMinPkEntryFlag=iMinPkEntryFlag+1;
120                     end
121                     iSizeMinPks=length(iRotMinPks);
122                     iSizeMinLocs=length(minLocs);
123                     iMinPeaksRemoved(iRemovePnt)=0;
124                 end
125             end
126             iRemoveElemFlag=0;
127         end
128     end
129 end
130
131 end
132
133 if pOutputPlot==1
134     % %plot the rotated data
135     plot(pVoltsRef(minLocs),iRotMinPks,'go');
136     hold on;
137 end
138 catch ME
```

```
139         iRotMinPks(1)=0;
140     end
141     minPksData=iRotMinPks;
142
143 end
```

Listing 5: Code to evaluate the min peaks and disregard min peaks we don't need

The function listing 5, calculates min peak points for a given data. The function is executed by passing the necessary parameters which it requires. These parameters are the following:

- pMaxPksData - this holds the max peak points found data.
- pMaxLocs - this holds the max peak points location found. The location is the position on the file currently being processed
- pCalDifData - This holds points data, for which we are going to carry out the analysis on.
- pMPP - this holds the min peak prominence value.
- pVoltsRef - this holds the raw data.
- pDoMinPointEval - this is a flag parameter was a future though, if failure to resolve the min peak points, then the min peak prominence values would slightly broaden.
- pOutputPlot - this is a flag for outputting results plot on screen.
- minPksData - this holds the results data, which is outputted by the function.

Once the function runs, it initialises the data variables at the top of the function, which are in line 5 to line of the listing. The comments for the above variables can be found in the code listing. Next the function evaluates the size of max peak and location data, this is stored in line 14 and 15. The comments can be found in the code listing for the above variables. Once established number of max points, so we increment the size of the max data. This value will be used as datum for min points we need to find. The data held for analysis in 'pCalDifData' is rotated 180 degrees and stored in the variable 'iRotDatapnts', the findpeak function is used with the following parameters, 'MinPeakProminence' - this lets function know a custom value needs to be applied and pMPP - custom value for the above parameter, and the findpeaks will output the following result and they are captured by the following variables

- minPks - min peak point found
- minLocs - min peak location points to the raw file data.
- minW - extra data outputted by the function which is not required.
- minP - same as above.

The output is still in rotated form, the next execution statement line number 28 is allocated to the variable 'rotMinPksData' and is used for testing and debugging. The statement at line number 29, rotates the back to its original form and is allocated to the variable 'iRotMinPks'. Line number from

30 to 34 are setting up debugging test point variable 'minPeaksPnts', allocating min location points to variable 'minPeakLoc', allocating size of min peaks/min locations collection to variable 'iSizeMinPks, iSizeMinLocs'. Line number from 43 to 54 are variables initialise with default values for the nested for-loop for filtering the unwanted min peak points from collection of points. These are as follows:

- iElemToRemove = 0; this variable is used to keep count of number min peak points removed.
- iCIsTolElm = 1; – check first point is for its validity.
- iRemoveElemFlag = 0; – it's set to 1 when valid min peak point is found to be removed.
- iMinElemProcessed = 0; – keeps a count of number min points have processed.
- iNxtStrtCnt=0; – this is only used for debugging proposes.
- iMinLocChksVal=4; – number of min peaks elements required data.

Line number 58 is a start of a try catch block. Line number 59 is the start of if statement block, which tests, do we sufficient number of min peak points to start the filtering process of the unwanted points. Line number from 62 to 129 is a nested for-loop block. The flow diagram shown below explains the whats happening in the nest for-loop.

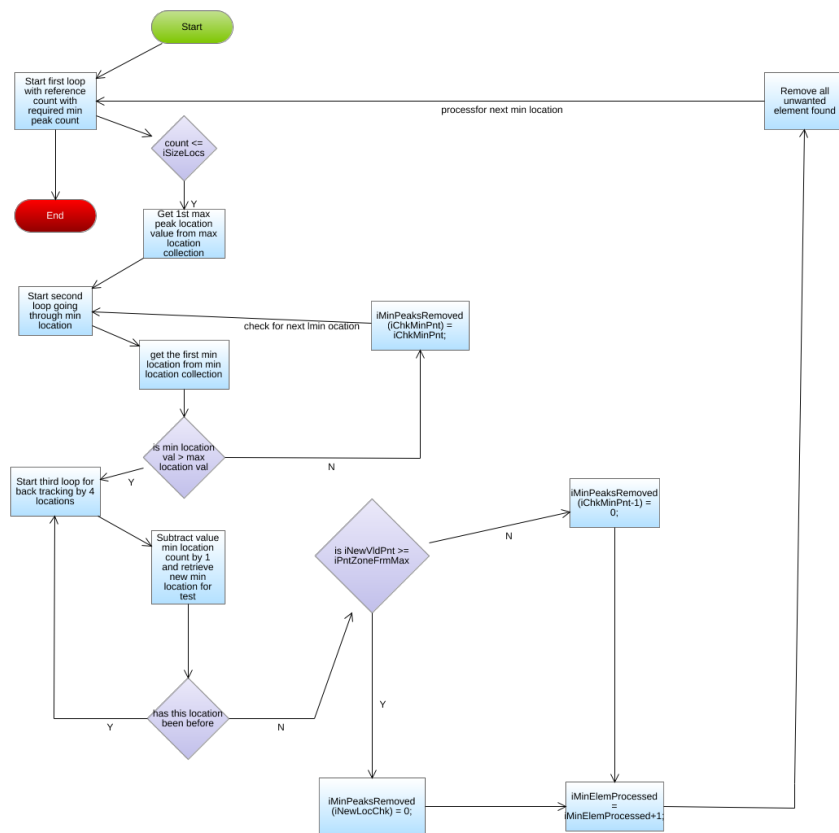


Figure 12: Flow diagram to explain what the for loop is doing at every process

Line numbers from 133 to 137 is the if block, it check for variable 'pOutputPlot' for value 1, it outputs the result to graph window. Lines from 138 to 140 is the catch block. If any problem occurs in processing of filtering of the unwanted min peaks data, variable 'iRotMinPks' set the first dimension to zero. The line number 141, set the variable 'minPksData' for all the valid min peak points.

```
1 function PeakLensData=EvaluatePeaksLengths(pFileNum , pMinPksData , pMaxPksData
    , pMaxLocs , pVoltsRef , pPeakLens)
2
3     PntsB4Zero=0;
4     PntsA4Zero=0;
5     [ PntsB4Zero , PntsA4Zero]=EvaluateMaxPeaksPostions(pMaxPksData ,
        pMaxLocs , pVoltsRef );
6
7     iSizeB4Pnts=0;
8     for iNumNeg=1:length(PntsB4Zero)
9         if PntsB4Zero(iNumNeg,2)>0
10             iSizeB4Pnts=iSizeB4Pnts+1;
11
12         end
13     end
14     iSizeA4Pnts=0;
15     for iNumPos=1:length(PntsA4Zero)
16         if PntsA4Zero(iNumPos,2)>0
17             iSizeA4Pnts=iSizeA4Pnts+1;
18
19         end
20     end
21     iTotalPnt=iSizeB4Pnts+iSizeA4Pnts;
22
23     iSizeMinPks=length(pMinPksData);
24     iSizeMaxPks=length(pMaxPksData);
25     iMinPntRq=iSizeMaxPks+1;
26
27
28     %Min to Point to skip
29     iSkipPoint=0;
30     if iSizeMaxPks==3
31         iSkipPoint=2;
32     elseif iSizeMaxPks==4
33         iSkipPoint=3;
34     elseif iSizeMaxPks==5 || iSizeMaxPks==6
35         iSkipPoint=4;
36     end
```

```
37
38     iNegPntsUsed=0;
39     iPosPntsUsed=0;
40     iPntsProcessed=0;
41     iStrtPos=1;
42     if iSizeMinPks>=iTotalPnt
43         %length of the negative point
44         if iSizeB4Pnts>=3
45             iNegPntsUsed=2;
46             iStrtPos=2;
47             for iNegPnts=1:iNegPntsUsed
48                 iMinPnt=pMinPksData(iStrtPos);
49                 iMaxPnt=pMaxPksData(iStrtPos);
50                 iPntsProcessed=iPntsProcessed+1;
51                 pPeakLens(pFileNum,iPntsProcessed)=iMaxPnt-iMinPnt;
52                 iStrtPos=iStrtPos+1;
53             end
54         else
55             for iNegPnts=1:iSizeB4Pnts
56                 iMinPnt=pMinPksData(iNegPnts);
57                 iMaxPnt=pMaxPksData(iNegPnts);
58                 iPntsProcessed=iPntsProcessed+1;
59                 pPeakLens(pFileNum,iPntsProcessed)=iMaxPnt-iMinPnt;
60
61             end
62         end
63         if iPntsProcessed<2
64             iPntsProcessed=iPntsProcessed+1;
65         end
66         if iSizeA4Pnts>=3
67             iPosPntsUsed=2;
68             for iPosPnts=1:iPosPntsUsed
69                 iMaxLoc=PntsA4Zero(iPosPnts+1,2);
70                 %iMinPnt=pMinPksData(iSkipPoint+iPosPnts);
71                 iMinPnt=pMinPksData(iMaxLoc+1);
72                 iMaxPnt=pMaxPksData(iMaxLoc);
73                 iPntsProcessed=iPntsProcessed+1;
74                 pPeakLens(pFileNum,iPntsProcessed)=iMaxPnt-iMinPnt;
75             end
76         else
77             if iSizeA4Pnts==1
```

```

79         iMaxLoc=PntsA4Zero(iSizeA4Pnts,2);
80
81         iMinPnt=pMinPksData(iSizeMinPks);
82         iMaxPnt=pMaxPksData(iMaxLoc);
83
84         iPntsProcessed=iPntsProcessed+1;
85         pPeakLens(pFileNum,iPntsProcessed)=iMaxPnt-iMinPnt;
86
87
88     else
89         for iPosPnts=1:iSizeA4Pnts
90             iMaxLoc=PntsA4Zero(iPosPnts,2);
91
92             %iMinPnt=pMinPksData(iSkipPoint+iPosPnts)
93             iMinPnt=pMinPksData(iMaxLoc+1);
94             iMaxPnt=pMaxPksData(iMaxLoc);
95
96             iPntsProcessed=iPntsProcessed+1;
97             pPeakLens(pFileNum,iPntsProcessed)=iMaxPnt-iMinPnt;
98         end
99     end
100 end
101
102 end
103 PeakLensData=pPeakLens;
104 end

```

Listing 6: Code to get the height of the peaks

The function in listing 6, calculates the lengths of the peaks between min and max peak points. To execute the function, it requires the six following parameters to be passed to it which are as follows:-

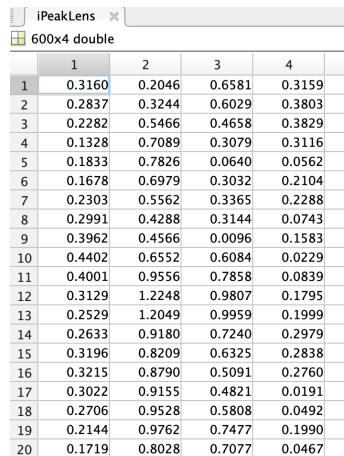
- pFileNum – current number of the file being processed.
- pMinPksData – collection of results of valid min peaks data
- pMaxPksData – collection of results of max peaks data
- pMaxLocs – collection of max peaks locations
- pVoltsRef – collection of the raw volts data.
- pPeakLens – collection for storing calculated peaks lengths
- PeakLensData – output results of calculated lengths.

lines from 3 to 4 are initialising the variables 'PntsB4Zero=0 and PntsA4Zero=0'. Lines number 5, is executing the function from listing 5 and from lines number 8 to 29, it goes through the returned results

from the function and it checks number of peaks before the zero and number of peaks after zero. Line number 21, it sums the value count for before zero peaks and after zero peaks, the value is store into variable 'iTotalPnt'. The Line numbers from 23 to 24, size of the collections pMinPksData and pMaxPksData, the values is store into the 'iSizeMinPks, iSizeMaxPks'. Next increments the 'iSizeMaxPks' and value is store to 'iMinPntRq'. From lines number 28 to 36, checks number of max peak point available and setup the skip points and values is stored in 'iSkipPoint'. Line number from 38 to 43, this block initailise the following variables with default values:

- iNegPntsUsed=0;
- iPosPntsUsed=0;
- iPntsProcessed=0;
- iStrtPos=1;

Lines from 42 to 102, its an if block. The if block check number of min peak points are greater or equal to total point found by the function 'EvaluateMaxPeaksPostions'. If the criteria is met, lines from 44 to 62 are executed to calculate the peak lengths for all peaks before the zero on x axis. First, check if number of min peaks before zero are greater or equal to three than lines number from 45 to 53 are executed. This block consists of for-loop and cycled through and values are taken from min/max peak data collection stored in variables 'iMinPnt, iMaxPnt'. iPntsProcessed variable is incremented, this holds count of points processed. The stored values in variables 'iMaxPnt and iMinPnt' are substracted and result is stored in the variable 'pPeakLens(pFileNum,iPntsProcessed)'. From lines numbers 54 to 61 is the else block of the if statement. This block has for-loop as describe apart from the parameter use by the for-loop and extracting the min/max peak data from the collection and the substracted result is stored in the variable 'pPeakLens(pFileNum,iPntsProcessed)' as describe above. From line number from 63 to 64 is an if block, which checks results have been process for before zero peaks. If the process points were less than two than the variable 'iPntsProcessed' is incremented. from line number from 66 to 100 is code block to evaluated peak lengths for after zero peaks, which is replication of the calculation for before zero peaks. The results is store in the variable 'pPeakLens(pFileNum,iPntsProcessed)'. The line number 103, the evaluated lengths are set to variable 'PeakLensData' which is passed out from the function. The output for the length of the data can be seen below in figure 13



	1	2	3	4
1	0.3160	0.2046	0.6581	0.3159
2	0.2837	0.3244	0.6029	0.3803
3	0.2282	0.5466	0.4658	0.3829
4	0.1328	0.7089	0.3079	0.3116
5	0.1833	0.7826	0.0640	0.0562
6	0.1678	0.6979	0.3032	0.2104
7	0.2303	0.5562	0.3365	0.2288
8	0.2991	0.4288	0.3144	0.0743
9	0.3962	0.4566	0.0096	0.1583
10	0.4402	0.6552	0.6084	0.0229
11	0.4001	0.9556	0.7858	0.0839
12	0.3129	1.2248	0.9807	0.1795
13	0.2529	1.2049	0.9959	0.1999
14	0.2633	0.9180	0.7240	0.2979
15	0.3196	0.8209	0.6325	0.2838
16	0.3215	0.8790	0.5091	0.2760
17	0.3022	0.9155	0.4821	0.0191
18	0.2706	0.9528	0.5808	0.0492
19	0.2144	0.9762	0.7477	0.1990
20	0.1719	0.8028	0.7077	0.0467

Figure 13: The output from the code for the length of the peak

To check if the values are correct we can manually calculate peaks, we can take the first 6 data files, and plot them individually and take the max peak value and subtract it from the min peak value for all the peaks we can see, and if we do this we get the following result

	Negative Peak 2	Negative Peak 1	Positive Peak 1	Positive peak 2
Data 1	0.316	0.205	0.658	0.316
Data 2	0.283	0.325	0.603	0.388
Data 3	0.228	0.547	0.465	0.328
Data 4	0.133	0.709	0.317	0.316
Data 5	0.181	0.783	0.286	0.250
Data 6	0.168	0.698	0.303	0.151
Data 7	0.230	0.556	0.333	0.199

Table 1: The manual calculation for the peaks from the first 7 $\frac{dI}{dV}$ against voltage graphs

The negative and positive peaks are can be seen in figure 9, the furthest blue from is the negative Peak 2 and the one closest to 0 is negative peak 1 same goes for the positive peaks. For all the positive peaks 2, the data is only a bit off, apart from 2 of the positive peaks in data 6 and 7 where they are incorrect, this is a success rate of $\frac{26}{28} \times 100 = 92\%$ which is good. We can do the following code to plot the graphs and separate the data into their respective parts

```

1
2
3 NH2 = iPeakLens (:,1)
4 NH1 = iPeakLens (:,2)
5 PH1 = iPeakLens (:,3)
6 PH2 = iPeakLens (:,4)
7
8 DiffH = ((PH2 - NH2)/PH2)*100
9 DiffH1 = ((PH1 - NH1)/PH1)*100
10

```

```
11 figure (1)
12 plot( AllB_field , DiffH , 'k-')
13 xlabel( 'B-Field (mT) ' )
14 ylabel( 'Percentage ' )
15 title( 'Difference of height between the 2nd positive Shapiro step and 2nd
        Negative Shapiro step ' )
16 figure (2)
17 plot( AllB_field , DiffH1 , 'k-')
18 xlabel( 'B-Field (mT) ' )
19 ylabel( 'Percentage ' )
20 title( 'Difference in height between the 1st positive Shapiro step and 1st
        negative Shapiro step ' )
21
22 figure (1)
23 plot( AllB_field , NH2, 'k-')
24 xlabel( 'B-Field (mT) ' )
25 ylabel( 'dI/dV (1/\Omega) ' )
26 title( 'Height of second negative peak against B-field (2nd Negative
        Shapiro Step) ' )
27 yline(0.3, '-.b', '0.3');
28 yline(0, '-.b', '0');
29 figure (2)
30 plot( AllB_field , NH1, 'k-')
31 xlabel( 'B-Field (mT) ' )
32 ylabel( 'dI/dV (1/\Omega) ' )
33 title( 'Height of first negative peak against B-field (1st Negative Shapiro
        Step) ' )
34 figure (3)
35 plot( AllB_field , PH1, 'k-')
36 xlabel( 'B-Field (mT) ' )
37 ylabel( 'dI/dV (1/\Omega) ' )
38 title( 'Height of first positive peak against B-field (1st Positive Shapiro
        Step) ' )
39 figure (4)
40 plot( AllB_field , PH2, 'k-')
41 xlabel( 'B-Field (mT) ' )
42 ylabel( 'dI/dV (1/\Omega) ' )
43 title( 'Height of second positive peak against B-field (2nd Positive
        Shapiro Step) ' )
44 yline(0.3, '-.b', '0.3');
```

Listing 7: Code to produce the graphs for the height against the magnetic field

From line 3 to line 6, it sorts the data into the following variables;

- NH2 - Negative height 2
- NH1 - Negative height 1
- PH1 - Positive height 1
- PH2 - Positive height 2

From line 11 to 44, it uses the variables to plot height and the difference in height as a percentage against the magnetic field and it will output into 6 figures. This can be seen in the result section.

5 Results

The code from section 4.2, produced the following graphs:

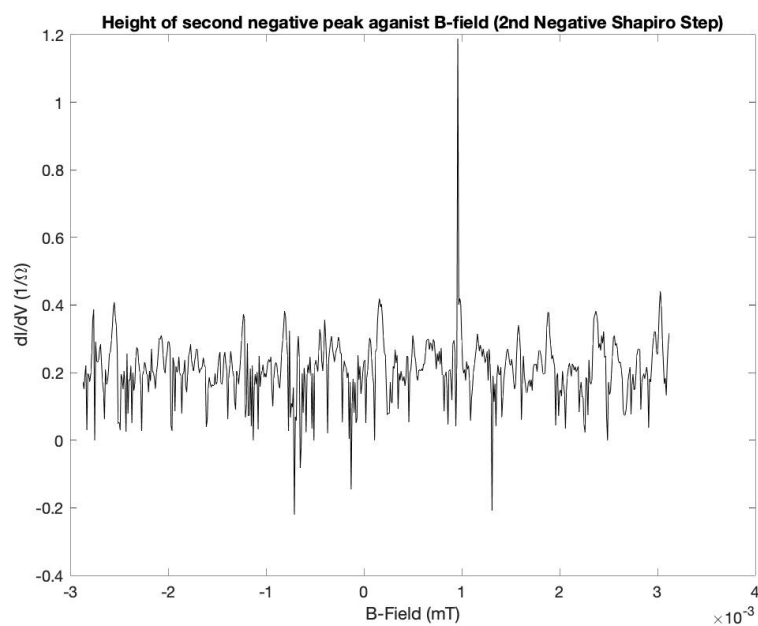


Figure 14: All the heights of the second negative Shapiro steps against the magnetic field

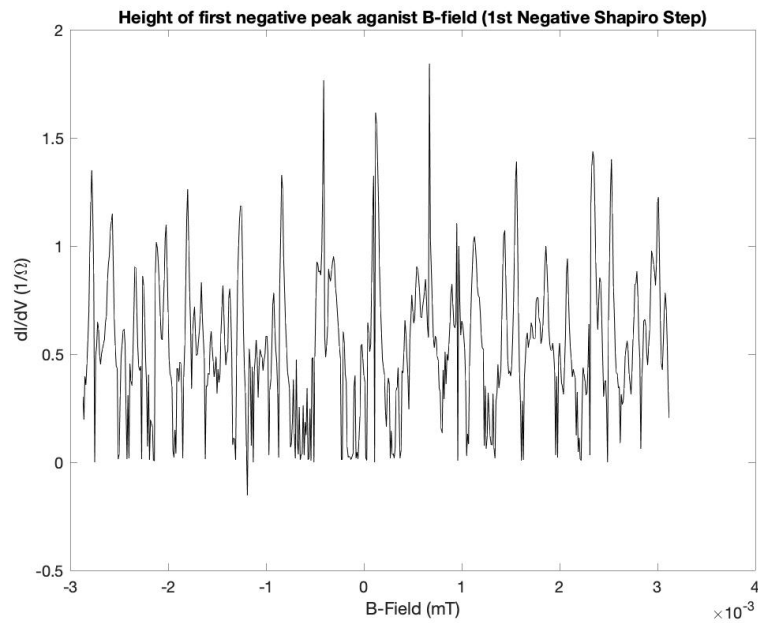


Figure 15: All the heights of the first negative Shapiro step against the magnetic field

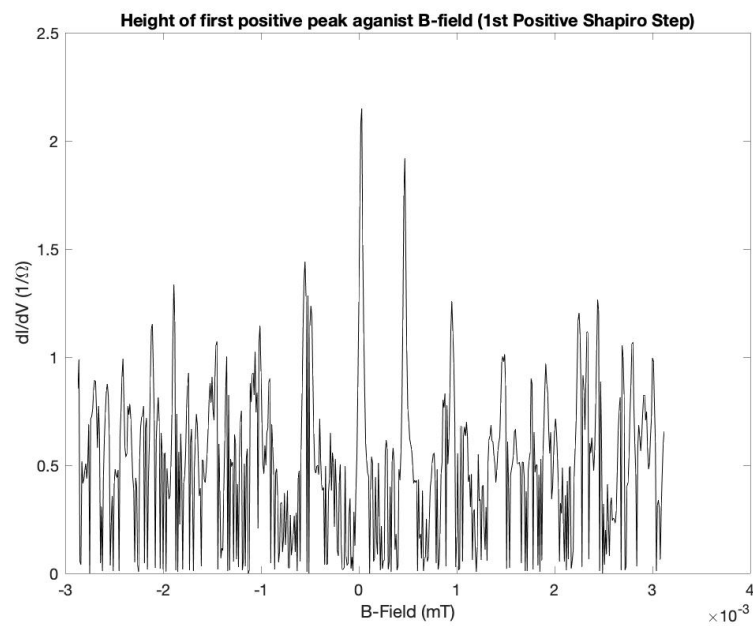


Figure 16: All the heights of the first positive Shapiro step against the magnetic field

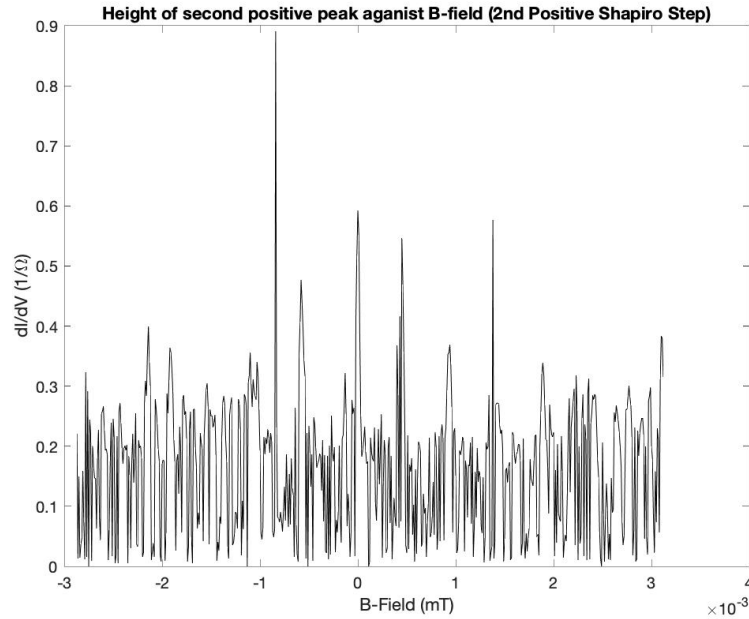


Figure 17: All the heights of the second positive Shapiro step

In figure 15 and 14 we can see at some points the data goes below 0 in the y-axis, this would mean that the min point is higher than the max point which can't happen so this is an error in the data. We can use the data from figure 14 to 17, to then calculate the difference in the heights of each Shapiro step, we do this by the following calculation:

$$DH1 = \left(\frac{NH1 - PH1}{NH1} \right) * 100 \quad (31)$$

$$DH2 = \left(\frac{NH2 - PH2}{NH2} \right) * 100 \quad (32)$$

where DH1 and DH2 are the difference in first Shapiro steps and second Shapiro step respectively, NH1 is the first peak on the negative side, NH2 is the second peak on the negative side, PH1 and PH2 are the first and second peak on the positive side respectively, this can be seen clearly from figure 9. Doing these calculation and plotting it against the magnetic field will give us the following graphs.

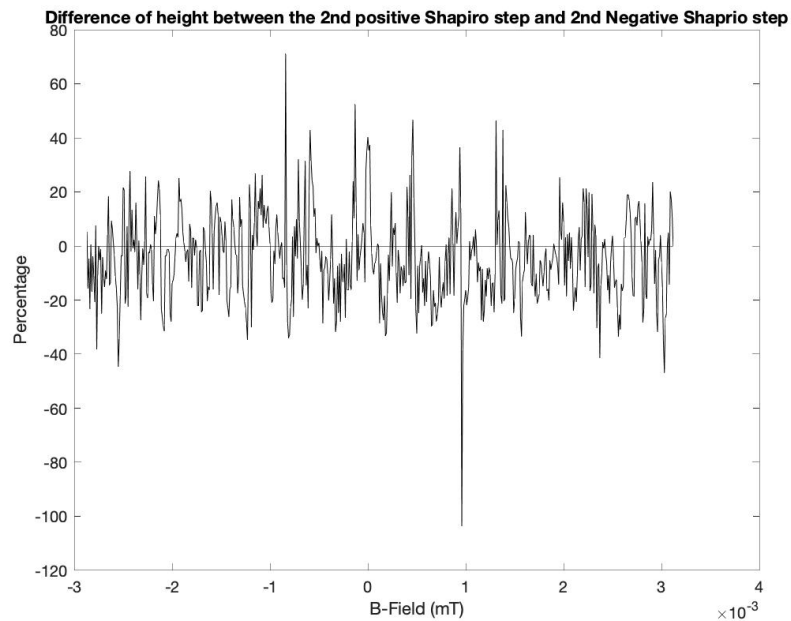


Figure 18: The difference between the All the first Shapiro Steps and All the first negative Shapiro steps against the magnetic field

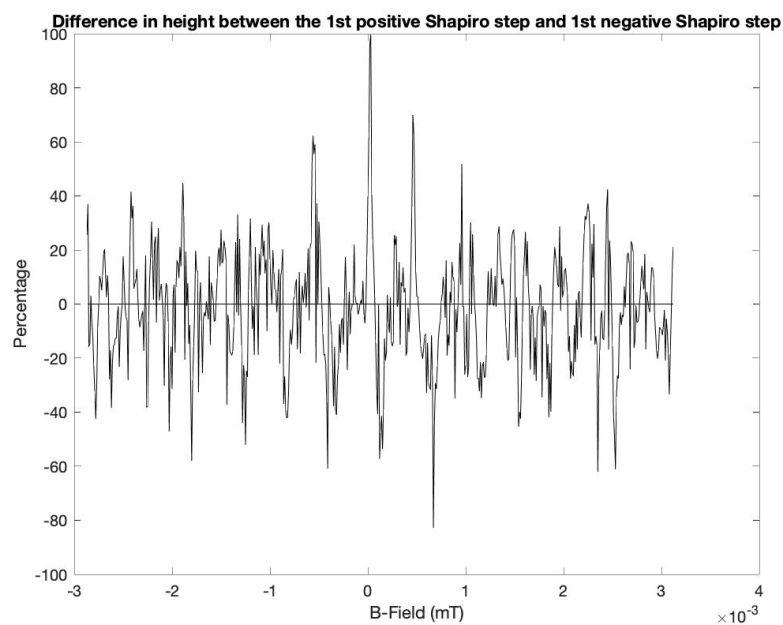


Figure 19: The difference between the All the second Shapiro Steps and All the second negative Shapiro steps against the magnetic field

From both figure 18 and 19, we can see that there are some negative percentages this indicates that the positive peak height is greater than the negative peaks height, from the graphs we can also see that none

of the peaks exceed the 100%, although in figure 16 there is a spike in the data where it reaches -100% this is because the values for the PH2 and NH2 when subtracted equaled the PH2 thus giving us -100%.

6 Discussion

All the graphs/data presented shows that the Shapiro steps vary with the magnetic field, but what also should happen is the the height of the second negative peak which is figure 14 and the height of the second positive peak which is figure 17 should be similar in terms of the their height, we can see that this is the case if you look at figure 20 and 21,

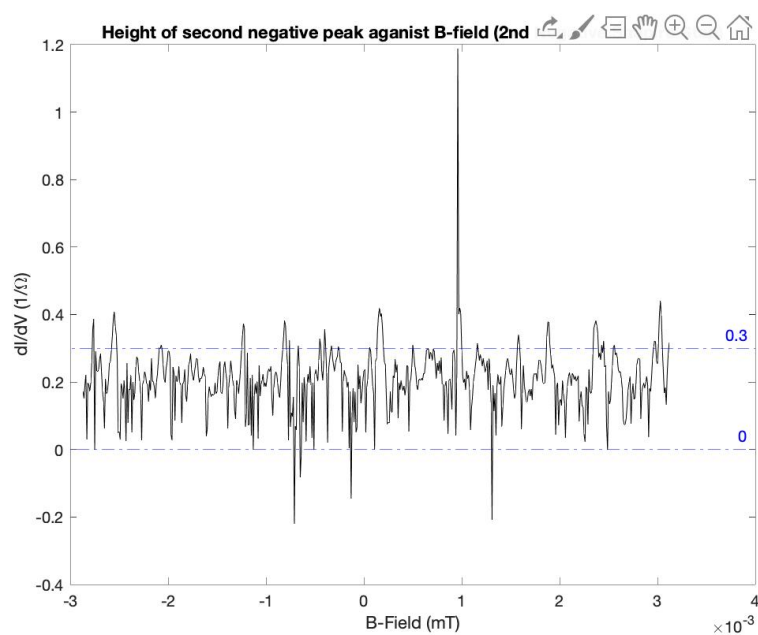


Figure 20: Same as figure 12, but a line at $y = 0.3$ and $y = 0$

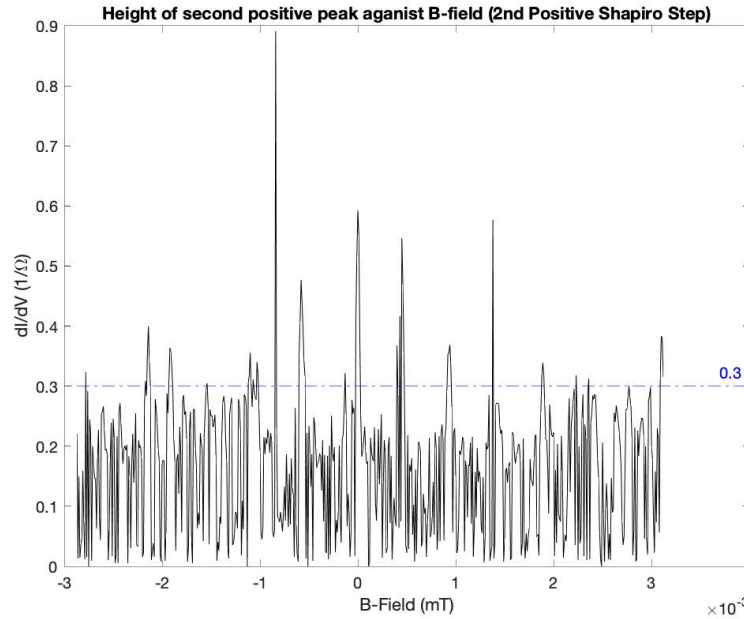


Figure 21: Same as figure 15, but a line at $y = 0.3$

Majority of the heights are between 0 and 0.3 for both graphs, while they are some outliers few a far between these are not taken into consideration, this can be seen as the in figure 18 percentage difference is relatively lower compared the figure 19. If we closely look at the height of the Shapiro steps the ones closest to the 0th voltage peak is higher than the furthest peaks, this indicates that when the order of Shapiro steps increases the height decays. Since the array was asymmetric this would imply that the difference in the heights can not be the same, but if the difference was to be zero then it would imply that there is an symmetric array, from figure 18 and 19 we can see that the difference in heights are different therefore the array was asymmetric.

7 Conclusion

In summary from the data presented we can confidently say that the number of Shapiro steps that are seen are 4, two on the positive side of the magnetic field and two on the negative side of the magnetic field, also we can see that the Shapiro steps varies with the magnetic field. The heights of the second positive and second negative are closely related, and we can see that as the order of the Shapiro steps increases the height decays, these results can also be seen in previous papers that have been highlighted. The asymmetry of the junction can be seen in the difference of the heights of the Shapiro steps

8 Acknowledgments

I would like to thank my supervisor Dr Boris Chesca.

References

- ¹D. van Delft and P. Kes, “The discovery of superconductivity”, American Institute of Physics. **NA**, 38–42 (2010).
- ²E. Hanno and F. M. C. N., “Meissner effect, diamagnetism, and classical physics—a review”, American Journal of Physics **80**, 164–169 (2012).
- ³A. A. Abrikosov, “Nobel lecture: type-II superconductors and the vortex lattice”, REVIEWS OF MODERN PHYSICS **76** (2004).
- ⁴R. Kleiner and W. Buckel, “Ideal diamagnetism, flux lines, and flux quantization”, in *Superconductivity*, edited by Wiley-VCH, Superconductivity (Wiley-VCH, Weinheim, Germany, 2016) Chap. 1, pp. 23–24.
- ⁵P. Chaddah, “Critical current densities in superconducting materials”, Sadhana **28**, 273–282 (2003).
- ⁶J. M. Martinis and K. Osborne, *Superconducting qubits and the physics of josephson junctions*, <https://web.physics.ucsb.edu/~martinisgroup/classnotes/finland/LesHouchesJunctionPhysics.pdf>, (accessed: 19.05.2020).
- ⁷H. Asai, Y. Ota, S. Kawabata, M. Machida, and F. Nori, “Theory of macroscopic quantum tunneling with josephson-leggett collective excitations in multiband superconducting josephson junctions”, Phys. Rev. B **89**, 224507 (2014).
- ⁸N. Turok, “On quantum tunneling in real time”, New Journal of Physics **16**, 063006 (2014).
- ⁹M. Moshe and R. G. Mints, “Shapiro steps in josephson junctions with alternating critical current density”, Phys. Rev. B **76**, 054518 (2007).
- ¹⁰R. Kleiner and W. Buckel, “Josephson currents”, in *Superconductivity*, edited by Wiley-VCH, Superconductivity (Wiley-VCH, Weinheim, Germany, 2016) Chap. 1.5.1, pp. 47–50.
- ¹¹J. Cox, B. Chesca, D. John, S. Savel’ev, and C. Mellor, “Vortex ratchets based on asymmetric arrays of josephson junctions”, Journal of Statistical Mechanics: Theory and Experiment **2019**, 114001 (2019).
- ¹²B. Ahlgren and D. Byström, *A study of josephson junction characteristics*, <https://www.diva-portal.org/smash/get/diva2:560204/FULLTEXT01.pdf>, (accessed: 19.05.2020).
- ¹³J. Blomgren and P. Magnelind, *The josephson effect*, <http://fy.chalmers.se/~delsing/LowTemp/Labbar/SQUIDlab-rev3.pdf>, (accessed: 19.05.2020).
- ¹⁴R. Feynman, *The schrödinger equation in a classical context: a seminar on superconductivity*, https://www.feynmanlectures.caltech.edu/III_21.html, (accessed: 19.05.2020).
- ¹⁵N. I. Guwahati, *Lec 15: josephson junctions, josephson equations*, <https://www.youtube.com/watch?v=i7cETh3AXQg>, (accessed: 19.05.2020).
- ¹⁶S. Levy, E. Lahoud, I. Shomroni, and J. Steinhauer, “The a.c. and d.c. josephson effects in a bose–einstein condensate”, Nature **449**, 579–583 (2007).
- ¹⁷R. Kleiner and W. Buckel, “The rcsj model”, in *Superconductivity*, edited by Wiley-VCH, Superconductivity (Wiley-VCH, Weinheim, Germany, 2016) Chap. 6.2, pp. 337–342.
- ¹⁸R. Kleiner and W. Buckel, “Josephson junctions under microwave irradiation”, in *Superconductivity*, edited by Wiley-VCH, Superconductivity (Wiley-VCH, Weinheim, Germany, 2016) Chap. 6.3, pp. 342–345.

- ¹⁹A. J. Gilberte, “Grain boundaries in high temperature superconducting ceramics”, *Philosophical Magazine* **86**, 2197–2243 (2006).
- ²⁰S. Chao, K. Xu, L. Hekang, Z. Yu-Ran, Z. Xu, L. Wuxin, G. Qiujiang, W. Zhen, R. Wenhui, H. Jie, F. Hui, F. Heng, Z. Dongning, W. Da-Wei, W. H., and Z. Shi-Yao, “Generation of multicomponent atomic schrödinger cat states of up to 20 qubits”, *Science* **365**, 574–577 (2019).
- ²¹O. A., L. H., K. A., S. G., W. T. T., E. S., B. H., Z. A. S., P. H., C. S., C. J., R. M., R. P., M. S., C. T., E. M., G. M., V. V., and L. M. D., “Generation and manipulation of schrödinger cat states in rydberg atom arrays”, *Science* **365**, 570–574 (2019).
- ²²F. Janicek, A. Cerman, M. Perny, I. Brilla, L. Marko, and S. Motycak, “Applications of superconducting quantum interference devices”, in 2015 16th international scientific conference on electric power engineering (epe) (2015), pp. 429–432.
- ²³S. Henry, *Superconducting quantum interference devices and their applications*, <https://stfc.ukri.org/files/superconducting-quantum-interference-devices-and-their-applications/>, (accessed: 19.05.2020).
- ²⁴W. H. B., W. P. H., and Y. T., “Terahertz responses of intrinsic josephson junctions in high T_C superconductors”, *Phys. Rev. Lett.* **87**, 107002 (2001).
- ²⁵S. Liou, W. Kuo, Y.-W. Suen, W. Hsieh, C. Wu, and C. Chen, “Shapiro steps observed in a superconducting single electron transistor”, *Chinese Journal of Physics- Taipei* **45**, 230–236 (2007).
- ²⁶H. Lin, W. Jian, and C. Moses, “Shapiro steps in the absence of microwave radiation”, (2011).

9 Appendix

9.1 Diary of meetings

Date: 04/10/19	Discussed the project
Date: 16/10/19	Asked question about the theory/ Shapiro steps and fiske steps
Date: 23/10/19	Asked question about the experiment, Josephson junction and the grain boundary
Date: 30/10/19	Asked question about the AC and DC Josephson effect
Date: 13/11/19	Asked questions about the RCSJ model
Date: 11/12/19	Showed the first iteration of code, suggestion made to smooth the data out
Date 4/03/20	Showed the first set of graphs/ Asked question on the analysis
Date: 11/03/20	Talked about how to check the data manually

9.2 Contribution

The data was collected was done prior to my involvement, but the code to analyse the said data was done by me as you can see in section 4.2, this includes calculating the difference in height as well as the Shapiro steps and plotting them against the magnetic field, this code was made for general use and can work for any file extension and any data set presented for any type of Josephson junction.