Josephson Junctions in Unconventional Superconductors

Björn H. Wehlin

Advisors:

Prof. David A. Meyer (Dept. of Mathematics)
Prof. Robert C. Dynes (Dept. of Physics)
Dr. Shane A. Cybart (Dept. of Physics)

May 22, 2015
Acknowledgements

First, I would like to express my sincerest gratitude towards my advisors David Meyer, Bob Dynes and Shane Cybart for the countless hours they have spent mentoring me in the research process, as well as their careful proof reading of this thesis. Without them this work would have never been possible.

Ethan Cho, Travis Wong and Meng Ma have provided me with ample amounts of data that has supported the theoretical predictions made herein.

I thank professors Jason Schweinsberg and Rayan Saab for a discussion on stochastic differential equations and numerical methods, respectively.

I am most grateful for the UC Scholars research stipend that I received during the summer of 2014, which funded this work in part. The support from Karen van Ness and Dr. David Artis of Academic Enrichment Programs during the UC Scholars Program, as well as various research conferences and events, has been much appreciated. On the same note, I would like to thank Terry Le with the UC San Diego math department for coordinating the honors program.

Last but not least I would like to thank my friends and family for their support.
To my parents
Preface

The goal of this work is to extend current models used to characterize Josephson junctions. In particular, extensions are considered for junctions in so-called unconventional superconducting materials. In the first chapter, we provide some necessary background information and establish some problems that arise in unconventional materials but are not present in their conventional counterparts, for which the theories were originally derived.

The effect that is given the most thorough treatment is an excess current $I_x$ which is present in several junction types. Due to the strong contribution of $I_x$ it is impossible to characterize many junctions using current models. We derive a new model in the second chapter and finish with numerical methods to use the model in the last chapter.

Based on our results, we see that the Ambegaokar-Halperin model [2], which is commonly used to characterize Josephson current-voltage data overestimates the figure of merit for Josephson junctions ($I_C R_N$) by around 100% in some cases.
Publications


(doi:10.1038/nnano.2015.76)
vi
# Contents

1 Introduction  
1.1 Preliminaries .......................... 3  
1.2 BCS Theory ................................. 4  
1.3 Josephson Junctions ........................ 7  
1.4 Tanaka-Kashiwaya Theory ................. 10  
1.5 Junction Types ............................. 11  

2 Electrical Transport  
2.1 Ideal Case ................................. 13  
2.2 Thermal Noise .............................. 16  
2.3 Excess Current ............................. 18  
2.4 Proposed Current-Voltage Model .......... 20  
2.5 Results .................................. 23  
2.6 Comparison of Models .................... 25  

3 Numerical Algorithm  
3.1 Offsetting ................................. 28  
3.2 Fit Function Switchover ................. 30  
3.3 $\delta$-Almost Convexity .................. 33
3.4 Residual Analysis ........................................... 35
3.5 Proposed Fitting Algorithm .............................. 41
3.6 Parallelization .............................................. 42

Appendices ................................................................ 45

A Implantation Studies ........................................... 47
  A.1 Junction Geometries ........................................ 49
  A.2 Prior Work .................................................. 49
  A.3 Results ...................................................... 50
  A.4 Future Work ................................................ 52

B Code: Residual Analysis ........................................... 57

C Code: Fitting Implementation ................................ 63
List of Symbols

\[ \Delta \] BCS energy gap

\[ \hbar \] Reduced Planck constant \((\hbar/2\pi = 1.055 \times 10^{-34} \text{ J s})\)

\[ \mathbb{R} \] Real numbers

\[ \mathbb{R}_+ \] Nonnegative real numbers

\[ \Omega_{\psi}^- \] Order parameter surface of \( \psi \) in pre-tunnel state

\[ \Omega_{\psi}^+ \] Order parameter surface of \( \psi \) in post-tunnel state

\[ \Phi \] Applied magnetic flux

\[ \Phi_0 \] Flux quantum

\[ B \] Magnetic field

\[ e \] Elementary charge \((1.602 \times 10^{-19} \text{ C})\)

\[ I_{\chi} \] Excess current

\[ I_C \] Josephson critical current

\[ I_s \] Supercurrent
$I_{qp}$  Quasiparticle current

$R_N$  Josephson normal-state resistance

$T_C$  Superconducting transition temperature

$V$  Time-averaged voltage across junction
Chapter 1

Introduction

One of the more curious phenomena of physics is that of superconductivity. In 1911, Heike Kamerlingh Onnes noticed that as a mercury sample is cooled below around 4.2 K, its electrical resistance drops abruptly to zero [25].\(^1\) In 1933, Walther Meißner and Robert Ochsenfeld made the further discovery that superconductors also expel magnetic fields [22]. It is for these two effects, the loss of electrical resistance and the Meißner-Ochsenfeld effect, superconductivity is perhaps best known. We shall, however, primarily concern ourselves with the former.

Despite physicists’ best efforts at conceiving a phenomenological theory that could explain superconductivity, it was not until 1957 that Bardeen, Cooper and Schrieffer [5] put forth their theory which has become known as the BCS theory of superconductivity.\(^2\) Although we make no attempt at a complete

---

\(^1\)For his discovery, Onnes was awarded the 1913 Nobel prize in physics.
\(^2\)Bardeen, Cooper and Schrieffer were awarded the 1972 Nobel prize in
description of the BCS theory, we will describe relevant parts. The key observation that the theory provides, and which Cooper theorized in 1956 [12], is that electrons in a superconductor form bound pairs, or so called Cooper pairs. The pair formation is somewhat involved and requires at the very least an understanding of the energy levels present in a crystal lattice. We will return to this in a later section and consequently postpone our further description of the BCS theory.

In what followed, Josephson proposed in 1962 that two superconductors separated by a thin insulating barrier should permit the flow of a tunneling supercurrent between the superconductors [17]. This configuration is called a Josephson junction. On a historical note, Bardeen first dismissed Josephson’s prediction [4] but it was later found correct as Anderson and Rowell made the first empirical observation of the Josephson effect in 1963 [3]. The Josephson junction will be our main focus within this thesis but we wish to establish some more background information before we continue our discussion on the topic.

In the years that followed Josephson’s prediction, many new materials were found to superconduct at temperatures nearing the limit of what the BCS theory predicted to be the fundamental limit. Then, in 1986, another breakthrough in superconductivity was made by Bednorz and Müller [6] with their discovery of materials that could superconduct at temperatures previously thought impossible. This new discovery was dubbed physics for the BCS theory.

---

3 Josephson shared the 1973 Nobel prize in physics for his prediction.
4 Bednorz and Müller received the 1987 Nobel prize in physics for their
1.1. PRELIMINARIES

High-temperature superconductivity (HTS) and subsequent developments in the field have lead to the discovery of materials with superconducting transition temperatures above 130 K, as reported in 1993 [27].

Today, Josephson junctions find wide use within many application areas ranging from interplanetary thermal imaging to biomagnetism and quantum computing. We wish to refer the interested reader to Clarke and Braginski [11] for more information.

1.1 Preliminaries

A $d$-dimensional crystal lattice, or Bravais lattice, is defined as an infinite, translation-invariant set of discrete points that is generated by a basis $\{a_1, a_2, \ldots, a_d\}$. Then, the vectors given by

$$\mathbf{r} = n_1 \mathbf{a}_1 + n_2 \mathbf{a}_2 + \cdots + n_d \mathbf{a}_d,$$

where $n_1, \ldots, n_d \in \mathbb{Z}$, are known as lattice sites. Unless stated otherwise, we shall restrict ourselves to the case where $d = 3$ as this is the case for actual crystals. From this, we can define the reciprocal lattice to be the lattice generated by

$$\mathbf{b}_1 = 2\pi \frac{\mathbf{a}_2 \times \mathbf{a}_3}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)},$$

$$\mathbf{b}_2 = 2\pi \frac{\mathbf{a}_3 \times \mathbf{a}_1}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)},$$

$$\mathbf{b}_3 = 2\pi \frac{\mathbf{a}_1 \times \mathbf{a}_2}{\mathbf{a}_1 \cdot (\mathbf{a}_2 \times \mathbf{a}_3)}.$$
This is the so-called momentum space, or k-space. Points in k-space give, for our purposes, the momenta of electrons at certain energy levels. A fuller discussion is given in, for example, [23] and [18].

For a system at 0 K, we define the Fermi level $E_F$ to be the highest occupied energy level of the electrons in the system. Then, in k-space, we see that a surface, which we will denote $\Omega_F$, defines all k-points that correspond to $E_F$. This surface is called the Fermi surface. In a free electron model (i.e., where the electrons are assumed to have no mutual interactions either between themselves or other constituents of the system), we see that $\Omega_F$ is a sphere as the energy of the electron $E \propto p^2$ is proportional to the square of its momentum. This spherical symmetry is also what the BCS theory assumes.

1.2 BCS Theory

The theoretical understanding of superconductors came with the BCS theory in 1957 [5]. We shall give a brief overview of some of the key concepts of the theory as they relate to our purposes. Furthermore, it is important to note that the BCS theory applies to conventional superconductors and not high-temperature superconductors. That being said, the theory sometimes gives useful predictions for these materials as well.

At the heart of the BCS theory is Cooper pairing; that is, the formation of bound states for electron pairs in a superconductor. Cooper showed that it is energetically favorable for two electrons with equal but opposite momentum and spin to form
such pairs in a superconductor [12]. Electrons are fermions, which means that they obey the Pauli exclusion principle that no two fermions in close proximity may occupy the same quantum state. However, paired electrons form spin singlets; in other words, they have vanishing spin. This is to say that the pairs undergo Bose-Einstein condensation. In the bosonic state, the Pauli exclusion principle no longer applies, and so the pairs can all jointly occupy the ground state. The electrons involved in the pairing process are those that lie close in energy to the Fermi energy; that is, the electrons that are close to the Fermi surface. The electrons that remain unpaired go into a quasiparticle state where they attain a different mass from normal electrons. We refer to Figure 1.1 for a schematic representation.

For our purposes, the perhaps most important property of the Bose-Einstein condensate is that it can be completely described in terms of a single wavefunction, which we choose to
CHAPTER 1. INTRODUCTION

represent here in its time-independent form,

\[ \psi(\mathbf{r}) = \sqrt{n_s(\mathbf{r})} e^{i\phi(\mathbf{r})}. \] (1.1)

Here, \( n_s(\mathbf{r}) \) is the density of states and so \( |\psi^*(\mathbf{r})\psi(\mathbf{r})| = n_s(\mathbf{r}) \) represents the superfluid density at \( \mathbf{r} \). We see also that the phase \( \phi(\mathbf{r}) \) of \( \psi \) is locally dependent. The superconducting wavefunction is normally called the order parameter. As we can map points in position space (i.e., \( r \)-space) into momentum space (i.e., \( k \)-space) we see that \( \psi \) also describes the momentum-space representation of the superconducting ground state.

Like wavefunctions for atomic electron orbitals give rise to symmetries, so does the superconducting order parameter, only here the symmetry arises in \( k \)-space instead of \( r \)-space. In a conventional superconductor, or BCS superconductor, the order parameter is isotropic, or \( s \)-wave using the same notation as that which is used for electron orbitals. BCS superconductors are therefore called \( s \)-wave superconductors. In high-temperature superconductors, however, the symmetry is more complex. While it is still an open problem what exactly the symmetry is, it is generally believed to be either \( d \)-wave or a combination of \( s \)-wave and \( d \)-wave [32].

The BCS energy gap \( \Delta \) is the energy required to break a Cooper pair. We can derive \( \Delta \) from the wavefunction in \( k \)-space in the direction of current flow by considering the length of a line segment connecting opposite points on the surface given by taking the expectation value of the momentum. It is clear that in an \( s \)-wave configuration, \( \Delta \) is isotropic. For other symmetries, it is not.
1.3 Josephson Junctions

Josephson proposed in 1962 [17] that two superconducting electrodes separated by a thin normal metal or insulating barrier should allow for the flow of a Cooper pair supercurrent $i_s(t)$ through the barrier. As pointed out earlier, a single wavefunction (see Equation 1.1) describes the whole superconducting system. However, when two superconductors are placed in close enough proximity, the Josephson effect dictates that the two phases will interlock. This gives a well-defined quantity $\delta(t) = \phi_1(t) - \phi_2(t)$, where $\phi_1, \phi_2$ are the spatially dependent phases on each side of the barrier. Josephson then predicted that the supercurrent would take the form

$$i_s(t) = I_c \sin \delta(t)$$

(1.2)

for a critical current $I_c$. Equation 1.2 is known as the first Josephson relation. Letting $I_b$ be a bias current applied through the junction, we obtain a zero potential difference for $|I_b| < |I_c|$ and develop a voltage outside this range. Several successful models to predict the junction current-voltage characteristics exist and we will return to this topic in a later chapter.

We will describe the tunneling process in terms of a pre-state and a post-state. We say that the tunneling pair is in its pre-state before it tunnels and in its post-state after the tunneling process has completed. From conservation of transverse momentum, the post-state must be equivalent to the pre-state along the transverse direction in k-space. Although k-space consists of discrete lattice points, we will make the approximation that we
can describe the ground state order parameter as a smooth surface $\Omega_\psi$ where each point on $\Omega_\psi$ corresponds to the ground state energy. We will call $\Omega_\psi$ the order parameter surface associated with the order parameter $\psi$.

Ambegaokar and Baratoff [1] derive a relationship between $I_c$ and temperature, $T$, for a given $\Delta$ in an isotropic configuration (see Figure 1.2). It is of interest to extend this result to superconductors with anisotropic order parameters. We introduce the notation $\Omega_\psi^\rightarrow$ to mean the order parameter surface of $\psi$ in the pre-state, and $\Omega_\psi^\leftarrow$ to be that of the post-state. It follows
1.3. JOSEPHSON JUNCTIONS

(a) Isotropic order parameter. Each pre-state has exactly one allowable post-state.

(b) Anisotropic order parameter with $\Omega^\leftarrow_\psi = \Omega^\rightarrow_\psi$. Multiple post-states are available for each pre-state.

(c) Anisotropic order parameter with $\Omega^\leftarrow_\psi = \mathcal{T}\Omega^\rightarrow_\psi$, where $\mathcal{T}$ is a rotation about the origin.

Figure 1.3: Schematic representation of tunneling in $k$-space. As electron pairs are tunneling through the barrier, the pre-state (left) must match the post-state (right).

that we must satisfy\textsuperscript{5}

$$\nabla\Omega^\rightarrow_\psi(k_1) = \nabla\Omega^\rightarrow_\psi(k_2)$$

(1.3)

for tunneling between the points $k_1$ and $k_2$ to be allowed. We will let $\mathcal{T}$ be a linear transformation that operates on an order parameter surface. By doing so, we may take

$$\Omega^\leftarrow_\psi = \mathcal{T}\Omega^\rightarrow_\psi$$

(1.4)

as a special case. The above discussion is incorporated into Figure 1.3.

\textsuperscript{5}We use $\nabla$ as the gradient operator.
1.4 Tanaka-Kashiwaya Theory

A theory for $d$-wave Josephson junctions that is in line with the more general theory eluded to herein has been proposed by Tanaka and Kashiwaya [31]. In this theory, two angles $\alpha$ and $\beta$ are defined and give rise to, in our language, two order parameter surfaces $\Omega_\psi^+ = R_\alpha(Sd_{x^2-y^2})$ and $\Omega_\psi^- = R_\beta(Sd_{x^2-y^2})$, where $Sd_{x^2-y^2}$ is the surface of a $d_{x^2-y^2}$ order parameter, and $R_\alpha$ and $R_\beta$ are rotation matrices rotating by $\alpha$ and $\beta$ degrees along the material’s $c$-axis, respectively (see Figure 1.4.)

It is assumed that the tunneling process is equivalent to that described in Ambegaokar-Baratoff [1] pointwise along each tunneling direction. The tunneling probabilities are derived using the Bogoliubov equation and a family of functions for $I_C(T)$ is derived. The results are plotted for some values of $\alpha$, $\beta$ in Figure 1.5.

An interesting research direction would be to test the Tanaka-Kashiwaya theory in high-$T_C$ junctions.
1.5. JUNCTION TYPES

Several types of Josephson junctions exist. Broadly, these can be classified as SNS, SIS and SS'S junctions. The letters refer to different material types, where S is a superconductor, N is a normal metal, I is an insulator and S’ is a different-\( T_C \) superconductor.\(^6\) An SNS junction consists of a superconductor (S) next to a normal-metal (N) barrier next to a superconductor (S), and so

---

\(^6\)We use the designation “normal metal” for any material that is not superconducting nor insulating.
on. Using this terminology, we can further describe material systems such as NS interfaces (normal-metal/superconductor interfaces) and even more complicated structures such as SS’IS’S junctions, etc.

Several junction fabrication techniques exist, including grain boundary [8], step edge [15], ramp edge [16] and ion damage [37] processes. More information about ion damage junctions can be obtained in Appendix A.

It has remained a long-standing problem to fabricate reproducible Josephson junctions in high-\(T_C\) materials. However, in a recent research effort, of which the present author was part, reproducible Josephson junctions in \(\text{YBa}_2\text{Cu}_3\text{O}_{7-\delta}\) have been shown possible using a focused ion beam technique [14].
Chapter 2

Electrical Transport

It is of prime experimental interest to determine the electrical transport properties of Josephson junctions. Much insight can be gained from fitting current-voltage models to experimental data. We will begin this chapter by giving a brief review of some current models and we then move on to extend these models to incorporate previously unaccounted for parameters. Common to all models described herein, however, is that the Josephson junction is being biased by a constant current $I_b$. That is, the junction lies in series with a current source.

2.1 Ideal Case

Ideal Josephson junctions are well-described by the Resistively-Capacitively Shunted Junction (RCSJ) model independently proposed by Stewart [29] and McCumber [21]. In this model, a junction $JJ$ described by the Josephson relation $I_s = I_C \sin \theta(t)$
Figure 2.1: Equivalent circuit used in the Stewart-McCumber (RCSJ) model [29, 21].

is connected in parallel with a resistor $R_N$ and a capacitor $C$ (see Figure 2.1). Conservation of energy together with the second Josephson relation then gives

\[
V(t) = \frac{\hbar}{2e} \frac{\partial \theta}{\partial t},
\]

(2.1)

\[
C \frac{dV}{dt} = I_b - \frac{V}{R_N} - I_C \sin \theta(t).
\]

(2.2)

Combining Equations (2.1) and (2.2) one arrives at the second-order ODE

\[
\frac{\hbar C}{2e} \frac{\partial^2 \theta}{\partial t^2} + \frac{\hbar}{2e R_N} \frac{\partial \theta}{\partial t} = I_b - I_C \sin \theta(t).
\]

(2.3)

An additional parameter $\beta_C = \left( \frac{2e}{\hbar} I_c R_N^2 \right) C$, known as the Stewart-McCumber parameter, is used to describe the capacitance of the junction. In most high-$T_C$ junctions, $\beta_C \ll 1$ leading to the second-order term in Equation 2.3 vanishing [19]. Without the capacitive term, the RCSJ model is referred to as the RSJ (Resistively-Shunted Junction) model.
2.1. IDEAL CASE

Figure 2.2: Plots of $I-V$ characteristics under the RCSJ model using the reduced units $\eta = V/I_CR_N$ and $\alpha = I_b/I_c$. The curve with $\beta_C = 0$ corresponds to the RSJ model and is described by Equation (2.3). (Adapted from Ref. [21])

The RSJ model can be solved analytically and the time-averaged voltage $\overline{V}$ takes the form

$$\eta(\alpha) = \begin{cases} 
0 & 0 < \alpha < 1, \\
(\alpha^2 - 1)^{1/2} & \alpha \geq 1,
\end{cases}$$

when written in the reduced units $\eta = \overline{V}/I_C R_N$ and $\alpha = I_b/I_C$.

Plots of the RSJ and RCSJ model are shown in Figure 2.2.
2.2 Thermal Noise

In junctions operating at higher temperatures, such as SNS junctions, thermal noise, or Johnson-Nyquist noise, can become an appreciable component of the overall transport process. To account for this phenomenon, the Ambegaokar-Halperin model [2] can be used to model the current-voltage characteristics. The effect can be seen in Figure 2.4. The Ambegaokar-Halperin model extends the RCSJ model by adding a stochastic noise current source (see Figure 2.3) so that Equation 2.2 becomes

$$C \frac{dV}{dt} = I_b - I_C \sin \theta - \frac{V}{R} + \tilde{L}(t),$$

(2.5)

where $\tilde{L}(t)$ is Gaussian noise with $\langle \tilde{L}(t+\tau)\tilde{L}(t) \rangle = 2R_N^{-1}T_N \delta(\tau)$, with $T_N$ being the noise temperature and $\delta$ being the delta function.

We will restate a summarized form of the derivation in Ref. [2]. Introducing the unitless parameters $\gamma = hI_C/ek_BT_N$, $x = I_b/I_c$ and $v = V/I_CR_N$, we can then transform Equations (2.1)
2.2. THERMAL NOISE

Figure 2.4: Current-voltage characteristics under the Ambegaokar-Halperin model. A higher noise temperature corresponds to a lower value of $\gamma$, and in the $\gamma \to \infty$ limit, the Ambegaokar-Halperin model becomes the RSJ model. (Adapted from Ref. [2])

and (2.5) into the form

$$\frac{d\theta}{dt} = \frac{p}{M},$$

(2.6)

$$\frac{dp}{dt} = \frac{dU}{d\theta} \eta - L(t),$$

(2.7)

where $p = (hC/2e)V$, $M = (h/2d)^2C$, $L = (h/2e)\tilde{L}$ and $U =$
$-\frac{1}{2}\gamma T_N (x\theta + \cos \theta)$. The problem is then analogous to Brownian motion by a particle of mass $M$ in a potential $U$ and admits a solution using a Focker-Planck equation. In the end, we arrive at the relationship

$$v = \frac{4\pi}{\gamma} \left\{ (e^{\pi\gamma x} - 1)^{-1} \int_0^{2\pi} d\theta \int_0^\theta d\theta' \frac{f(\theta)}{f(\theta')}ightharpoonup + \int_0^{2\pi} d\theta \int_0^{2\pi} d\theta' \frac{f(\theta)}{f(\theta')} \right\}^{-1}, \quad (2.8)$$

where $f(\theta) = \exp[-U(\theta)/T_N]$.

### 2.3 Excess Current

In certain junctions, an excess current becomes apparent. This current is in addition to the Josephson current $I_s$ and the quasi-particle current $I_{qp}$.

To make experimental assessments about the excess current, we may use the fact that $I_C$ modulates with magnetic flux. The $I_C$-flux relationship can be modeled in accordance with

$$I_C(\Phi) = I_C(0) \left| \frac{\sin(\pi\Phi/\Phi_0)}{\pi\Phi/\Phi_0} \right| = I_C(0) \text{sinc} |\Phi/\Phi_0|, \quad (2.9)$$

where $\Phi_0 = h/2e$ is the flux quantum (see for example Ref. [34]). Equation (2.9) is plotted in Figure 2.5 Given that $I_C(\Phi)$ has nodes at $n\Phi_0$, $n = \pm 1, \pm 2, \ldots$, and that $\Phi = B \cdot A$, where $A$ is the cross-sectional area of the junction, by introducing a magnetic field of strength $B = \Phi_0/A$ we can drive $I_C$ to zero.
2.3. **EXCESS CURRENT**

Likewise, we can modulate $I_C$ to its maximum by removing the magnetic field.

In reality, the $I_C$-flux relationship can be different, so we denote by $\Phi_{\text{max}}$ the point at which $I_C$ takes on its maximal value and by $\Phi_{\text{min}}$ the point at which $I_C$ vanishes.

If current is being swept at a high rate while magnetic field is being swept at a low rate, we can obtain current-voltage envelopes corresponding to full junction characteristics versus completely suppressed Josephson current. That is, the outer envelope corresponds to $\Phi = \Phi_{\text{max}}$ and gives the full junction characteristics, while the inner envelope corresponds to a current-
voltage characteristic composed of the quasiparticle current $I_{qp}$ and the excess current $I_x$ alone. We refer to Figures 2.6 and 2.7.

We leave it open to give a form for $I_x$ as a function of voltage and simply take it to be constant for now. The excess current is well-described for an NS interface by Blonder-Tinkham-Klapwijk [7] with subsequent extensions to SNS attempted by Octavio-Blonder-Tinkham-Klapwijk [24]. Seed-Vittoria-Widom [28] extends to RSJ model to include a constant excess current but does not account for thermal noise. Furthermore, none of the models described earlier account for the current-phase relationship $I_s(\theta)$ being different from the first Josephson relation $I_s(\theta) = I_C \sin \theta$. We therefore propose a current-voltage model comprising all the effects discussed.

### 2.4 Proposed Current-Voltage Model

We will now move on to introduce a new current-voltage model that takes all the three additional effects—excess current, non-linear resistance, and thermal noise—into account. By partitioning the current into three components $I_s$, $I_{qp}$ and $I_x$, the supercurrent (or Josephson current), quasiparticle current, and excess current, respectively, we arrive at the relationship

$$V(t) = \frac{\hbar}{2e} \frac{\partial \theta}{\partial t},$$

(2.10)

$$C \frac{dV}{dt} = I_b - I_{qp} - I_s - I_x - \tilde{L}(t).$$

(2.11)

In a first approximation, $I_s = I_C \sin \theta(t)$ is usually assumed, but we will allow for $I_s$ to be any periodic function. That is, we
2.4. PROPOSED CURRENT-VOLTAGE MODEL

Figure 2.6: Current (blue) is swept at a high rate which gives rise to voltage curves (red). At the same time magnetic field (black) is being swept at a low rate in order to modulate $I_C$.

demand

$$I_s(t + T) = I_s(t)$$ (2.12)
Figure 2.7: Plot of current versus voltage for the setup in Figure 2.6. The outer envelope, where $\Phi = \Phi_{\text{max}}$ gives a current-voltage characteristic including $I_s$, $I_{qp}$ and $I_x$. The inner envelope, where $\Phi = \Phi_{\text{min}}$ has the Josephson current completely suppressed and thus gives a current-voltage characteristic containing only $I_{qp}$ and $I_x$. The excess current at zero voltage has been marked as $I_x$ on the graph.

for any time $t$ and some period $T$.

The quasiparticle current, $I_{qp}$ will be assumed to be of ohmic nature and thus

$$I_{qp} = \frac{V}{R_N}.$$  \hspace{1cm} (2.13)

With these assumptions taken together with Equation (2.10),
we can rewrite Equation (2.11) and obtain the following system.

\[
V(t) = \frac{\hbar}{2e} \frac{\partial \theta}{\partial t}, \quad (2.14)
\]

\[
\frac{\hbar C}{2e} \frac{\partial^2 \theta}{\partial t^2} + \frac{\hbar}{2eR_N} \frac{\partial \theta}{\partial t} = I_b - I_s - I_X - \tilde{L}(t) \quad (2.15)
\]

\[
\langle \tilde{L}(t + \tau)\tilde{L}(t) \rangle = 2R_N^{-1}T_N \delta(\tau) \quad (2.16)
\]

### 2.5 Results

Using the model with constant \( I_X \), and \( I_s = I_C \sin \theta \), we have obtained several fits that agree well with experimental observations. Although these fits are still preliminary, they lend a high level of credibility to the validity of the model. Some fits are shown in Figure 2.8.
Figure 2.8: Our model fit to $I$–$V$ data for several temperatures: 74 K, 76 K, 78 K and 80 K.
2.6.  COMPARISON OF MODELS

We fit both the Ambegaokar-Halperin model and our model with $I_s = I_C \sin \theta$ and constant excess current. In this case, the Ambegaokar-Halperin model is a special case of our model with $I_X = 0$. A summary of the fitting parameters is shown in Table 2.1. Figure 2.9 shows the fits for 75 K.

From these comparisons, we see that the Ambegaokar-Halperin model, assuming that our model is correct, overestimates $I_C R_N$ (the figure of merit for Josephson junctions) by 94.5% in the 75 K and 100.7% in the 70 K case.

<table>
<thead>
<tr>
<th>Model</th>
<th>$I_C$ (µA)</th>
<th>$R_N$ (Ω)</th>
<th>$I_X$ (µA)</th>
<th>$I_C R_N$ (µV)</th>
</tr>
</thead>
<tbody>
<tr>
<td>AH (75 K)</td>
<td>75.1117</td>
<td>0.80438</td>
<td>N/A</td>
<td>60.4183</td>
</tr>
<tr>
<td>New (75 K)</td>
<td>37.6309</td>
<td>0.82534</td>
<td>32.7587</td>
<td>31.0583</td>
</tr>
<tr>
<td>AH (70 K)</td>
<td>166.5003</td>
<td>0.64507</td>
<td>N/A</td>
<td>107.4043</td>
</tr>
<tr>
<td>New (70 K)</td>
<td>94.2622</td>
<td>0.67371</td>
<td>66.2678</td>
<td>53.5054</td>
</tr>
</tbody>
</table>

Table 2.1: Comparison of fitted values under the Ambegaokar-Halperin (AH) model and our model (New).

2.6  Comparison of Models
Figure 2.9: Comparison of our proposed model (new) and the Ambegaokar-Halperin model (AH). We used the methods described in the next chapter to fit both models to experimental data (in this case, data for 75 K). Note that $T_N$ was held fixed at 100 K in the fits.
Chapter 3

Numerical Algorithm

Herein, we will describe a multi-step numerical algorithm to obtain fits to experimental data using the current-voltage model proposed in the preceding chapter.

To save notation, we may sometimes reuse variable names but we hope that the current state of a variable will be clear from the preceding discussion.

Let $D = \{(I_i, V_i)\}_{i=1}^{n}$ be a sequence of $n$ experimentally obtained current-voltage scatter points. We begin by re-indexing $D$ so that the $I$–$V$ points appear in ascending order in $I$. That is, $I_1 \leq I_2 \leq \cdots \leq I_n$. We then define a matrix

$$D = \begin{bmatrix} [D]_1 \\ [D]_2 \\ \vdots \\ [D]_n \end{bmatrix} = \begin{bmatrix} I_1 & V_1 \\ I_2 & V_2 \\ \vdots & \vdots & \vdots \\ I_n & V_n \end{bmatrix}.$$  

(3.1)
3.1 Offsetting

As our first step, we wish to center our data. That is, we will want to remove any offsets introduced by (systematic) experimental errors. This is most easily done by first considering the offset in voltage, $V_{\text{offs}}$. Here, we make use of the fact that the voltage of a Josephson junction remains at zero for an interval of current values $[-I_C, +I_C]$. But since we have not yet determined $I_C$ nor can we assume that there is no current offset $I_{\text{offs}}$, we will have to resort to further numerical schemes. We will, however, assume that the data are sufficiently centered in current so that there exists an interval $[-I_\ell, +I_\ell]$ over which the zero-voltage assumption holds.

To determine $I_\ell$, we find the first current-point with a non-negative value, say $I_m$, $\delta \leq m \leq n - \delta$, where $1 \leq \delta < n$ is some positive integer. Here, $\delta$ should be chosen such that the interval $[I_m - \delta, I_m + \delta]$ is fully contained in the zero-voltage region. One can either choose $\delta$ based on some knowledge of $D$, or $\delta$ can be determined programatically. If the latter is chosen we can determine two points $\delta_L, \delta_R$ where $I_j = I_m - \delta_L$ corresponds to the largest-index point at which $V_j \leq -V_{\text{th}}$ and $I_k = I_m + \delta_R$ corresponds to the smallest-index point at which $V_k \geq V_{\text{th}}$ for some threshold voltage $V_{\text{th}}$. That is,

$$
\delta_L = \max_{1 \leq j \leq n} \{ m - j : V_j \leq -V_{\text{th}} \},
$$

$$
\delta_R = \min_{1 \leq k \leq n} \{ k - m : V_k \geq V_{\text{th}} \}. \tag{3.2}
$$

We then let

$$
\delta = \left\lfloor K \min\{\delta_L, \delta_R\} \right\rfloor \tag{3.3}
$$
for some $K \in (0, 1)$. For most datasets, $K$ can be chosen to be in the vicinity of $1/5$ but special care will have to be taken for datasets with very few points.

With $\delta$ chosen, we can calculate the offset in voltage as

$$V_{\text{offs}} = \frac{1}{2\delta + 1} \sum_{i=m-\delta}^{m+\delta} V_i.$$  \hfill (3.4)

Next, we wish to treat the current offset $I_{\text{offs}}$. In order to do this, we must first apply the voltage offset obtained earlier by letting $[D]_{i2} = [D]_{i2} - V_{\text{offs}}$ for all $i = 1, \ldots, n$. This time, we will make use of the assumption that the current has been biased far enough that the junction has reached its normal state. That is, there exist integers $\gamma_L, \gamma_R$ for which $V_i \approx R_N I_i$ whenever $i \leq m - \gamma_L$ or $i \geq m + \gamma_R$. Let $\gamma = \min\{\gamma_L, \gamma_R\}$ so that $V_i \approx R_N I_i$ whenever $|i - m| \geq \gamma$.

There are several ways we can determine $\gamma$. The simplest of which is perhaps assuming that the smallest $i > m$ respectively largest $i < m$ for which $V_i \approx R_N I_i$ holds are equivalent to

$$\gamma_L := \max_{1<i<m} \left\{ i : V_i \leq -\lambda \max_{1\leq j \leq n} V_j \right\},$$

$$\gamma_R := \min_{m<i<n} \left\{ i : V_i \geq \lambda \max_{1\leq j \leq n} V_j \right\},$$  \hfill (3.5)

for some $\lambda \in (0, 1)$. Choosing $\lambda = 0.8 \pm 0.1$ has proven to be sufficient in most cases in our preliminary tests. A more sophisticated method might involve finding intervals with close-to-constant $dI/dV$, etc., but we have not found this added sophistication to be necessary in our testing.
To find $I_{offs}$ we will use the average value of the current evaluated at linearly interpolated points $I^-$ and $I^+$ corresponding to $I_{m-\gamma L}$ and $I_{m+\gamma R}$, respectively.

The current offset is thus

$$I_{offs} = \frac{I^- + I^+}{2}. \quad (3.6)$$

We adjust $D$ accordingly so that $[D]_{i1} = [D]_{i1} - I_{offs}$. At this point, we say that $D$ is centered.

### 3.2 Fit Function Switchover

In this section, we will consider continuous functions $f, g : E \subset \mathbb{R} \to \mathbb{R}$ defined on an open interval $E = (a, b)$ with the property that for any $\varepsilon > 0$, there exists $\xi \in E$ such that $|f(x) - g(x)| < \varepsilon$ for all $x > \xi$.

We then define a function $h : E \to \mathbb{R}$ given by

$$h(x) = \begin{cases} f(x) & a < x \leq \xi \\ g(x) & \xi < x < b \end{cases}. \quad (3.7)$$

Further, let $\{(x_i, y_i)\}_{i=1}^n$ be a sequence of points with $x_i \in E$ for all $i = 1, \ldots, n$ and $x_1 \leq x_2 \leq \cdots \leq x_n$. We denote by $i_{\xi}$ the smallest $i$ for which $x_i > \xi$. For any function $q : E \to \mathbb{R}$ we will use the notation

$$q(X) := \begin{bmatrix} q(x_1) & q(x_2) & \cdots & q(x_n) \end{bmatrix}^T \quad (3.8)$$

and take $X$ and $Y$ to be the column vectors of $x_i$’s and $y_i$’s, respectively.
Let $F$ be the space of functions from $E \to \mathbb{R}$. For a discrete set of points $P = \{p_1, \ldots, p_n\} \subset E$, we introduce the operator $\text{cost}_P : F \to \mathbb{R}_+$ defined as the worst-case computational evaluation time when evaluating the operand over all points in $P$. If $P = \{p_1\}$ is a singleton set, we use the shorthand $\text{cost}_{p_1}$ to mean $\text{cost}_\{p_1\}$.

We assume $\text{cost}_{x_i}(f) \geq \text{cost}_{x_i}(g)$ for all $x_i \in X$. The purpose of this section is to establish a method to evaluate $f$ over $P$ using $g$ as an approximation for points $p_i > \xi$.

The following results are easily seen to be true and will therefore be stated without proof.

**Proposition 1.** If $\alpha : E \to \mathbb{R}$ and $P_1, P_2 \subset E$, we have

$$\text{cost}_{P_1 \cup P_2}(\alpha) = \text{cost}_{P_1}(\alpha) + \text{cost}_{P_2}(\alpha) - \text{cost}_{P_1 \cap P_2}(\alpha). \quad (3.9)$$

**Proposition 2.** $\text{cost}_X(f) = \sum_{i=1}^{n} \text{cost}_{x_i}(f)$, and the same holds for $g$.

In a fitting procedure, it is clear that if $h$ is to be evaluated $m$ times, there will be $m$ values of $i_\xi$, denote them $i^j_\xi$, $j = 1, \ldots, m$. Suppose that $i^j_\xi \in [\ell, \ell + 1, \ell + u]$ where $1 \leq \ell \leq n$ and $0 \leq u \leq n - \ell$. We define $h_j$ as the $j$-th evaluation of $h$. Suppose further that $\text{cost}_{x_i}(f) \leq c_f$ and $\text{cost}_{x_i}(g) \leq c_g$ for some $c_f, c_g \in \mathbb{R}_+$ with $c_f \leq c_g$. Evidently, then, we may put bounds on the total computational cost.

**Theorem 1.** Let $T$ be the total worst-case evaluation time required to evaluate $h$ $m$ times. Then

$$T < m[(\ell + u)c_f + (n - \ell)c_g]. \quad (3.10)$$
CHAPTER 3. NUMERICAL ALGORITHM

Proof. For a fixed $j$, we have

$$\text{cost}_X(h) = \text{cost}_{\{x_1, \ldots, x_j\}}(f) + \text{cost}_{\{x_{i_\xi j+1}, \ldots, x_n\}}(g)$$

$$= \sum_{i=1}^{i_\xi j} \text{cost}_{x_i}(f) + \sum_{i=i_\xi j+1}^n \text{cost}_{x_i}(g)$$

$$\leq (i_\xi j - 1)c_f + (n - i_\xi j - 1)c_g$$

$$< i_\xi j c_f + (n - i_\xi j) c_g, \quad (3.11)$$

using Propositions 1 and 2. The result follows using the bounds on $i_\xi j$. \qed

Next, we are interested in the effect on the residuals.

**Theorem 2.** The error in the residual calculation introduced through fit function switchover is smaller than $(n - i_\xi)^{1/2} \varepsilon$. That is,

$$||h(X) - Y(X)||_2 < ||f(X) - Y(X)||_2 + (n - i_\xi)^{1/2} \varepsilon. \quad (3.12)$$

Proof. For brevity, we define

$$F_k := \sum_{i=1}^k |f(x_i) - Y(x_i)|^2. \quad (3.13)$$
3.3. \(\delta\)-ALMOST CONVEXITY

Then
\[
||h(x) - Y(x)||_2^2 = F_i + \sum_{i=i+1}^n |g(x_i) - Y(x_i)|^2
= F_i + \sum_{i=i+1}^n |g(x_i) - f(x_i) + f(x_i) - Y(x_i)|^2
\leq F_n + \sum_{i=i+1}^n |g(x_i) - f(x_i)|^2
< F_n + \sum_{i=i+1}^n \varepsilon^2
= F_n + (n - i - 1)\varepsilon^2
\]
\[
< F_n + (n - i)\varepsilon^2. \tag{3.14}
\]

It follows that
\[
||h(x) - Y(x)||_2 < \sqrt{F_n + (n - i)\varepsilon^2} \leq \sqrt{F_n + (n - i)^{1/2}\varepsilon}.
\tag{3.15}
\]

\[\Box\]

3.3 \(\delta\)-Almost Convexity

Let \(X, Y \in \mathbb{R}^m\) and \(f_\theta : \mathbb{R} \rightarrow \mathbb{R}\) be a function parameterized by \(p\) parameters in the form of a vector \(\theta \in \Theta \subset \mathbb{R}^p\) where \(\Theta\) is a convex subset of \(\mathbb{R}^p\). Let
\[
Y_\theta = \begin{bmatrix} f_\theta(x_1) & \cdots & f_\theta(x_m) \end{bmatrix}^T \tag{3.16}
\]

where \(x_i\) is the \(i\)-th component of \(X\).
We seek to solve
\[
\begin{aligned}
\text{minimize} \quad & \mathcal{R}(\theta) := ||Y_\theta - Y||_2 \\
\text{s.t.} \quad & \theta \in \Theta
\end{aligned}
\] 
(3.17)

In other words we are fitting a function \( f \) to experimental data \((X, Y)\) over a parameter space \( \Theta \). Notice that we have made no assumptions on \( \mathcal{R}(\theta) \) being convex or even continuous. However, the theory presented herein will pertain to functions that are almost convex. In order to lend further meaning to this statement we will now give some definitions.

**Definition 1.** Let \( C \subseteq \mathbb{R}^p \) be a compact, convex set and \( E \subseteq C \). We then say that \( C \) is an approximation domain of \( E \).

**Definition 2.** Let \( C \) be an approximation domain of a set \( E \in \mathbb{R}^p \). If \( f : E \to \mathbb{R} \) is bounded, and \( g : C \to \mathbb{R} \) is a convex function with the property that \( g(x) \geq f(x) \) for all \( x \in E \), we say that \( g \) is a convex lid of \( f \) over \( C \).

We denote by \( \mathcal{C}(f, C) \) the set of all convex lids of \( f \) over an approximation domain \( C \).

**Theorem 3.** Let \( C \) be an approximation domain of \( E \). Any function \( f : E \subseteq \mathbb{R}^p \to \mathbb{R} \) bounded above on \( E \cap C \) possesses a convex lid over \( C \). (That is, \( \mathcal{C}(f, C) \) is nonempty for \( f \) bounded above on \( E \cap C \).)

**Proof.** Suppose \( f \) is bounded above by \( \alpha \in \mathbb{R} \). Then, \( g : C \to \mathbb{R} \), \( g(x) := \alpha \) is a convex lid of \( f \). \( \square \)

To induce an order on convex lids, we will use the supremum metric \( d_\infty(f, g) = \sup_{x \in E} |g(x) - f(x)| \), where \( g \in \mathcal{C}(f, C) \) and \( C \) is an approximation domain of \( E \).
**Definition 3.** We say a bounded function \( f : E \subset \mathbb{R}^p \rightarrow \mathbb{R} \) is at most \( \delta \)-almost convex if

\[
\inf_{g \in \mathcal{C}(f, C)} d_\infty(f, g) \leq \delta
\]

for some approximation domain \( C \) and \( \delta \in \mathbb{R}_+ \). If equality holds, we say \( f \) is \( \delta \)-almost convex. For a specific \( \delta = \delta_0 \), we may say \( f \) is (at most) almost convex with \( \delta = \delta_0 \).

**Remark.** A function is convex if and only if it is almost convex with \( \delta = 0 \).

### 3.4 Residual Analysis

A MATLAB program was written to calculate the residuals \( R \) for various data sets. This program is presented in Appendix B. The approach is to divide the fitting process into two steps. In the first step, \( I_\chi \) and \( T_N \) are fixed and then a standard constrained least squares algorithm is employed to fit for \( I_C \) and \( R_N \), giving the full parameterization.

On first inspection \( R \) appears to be convex (see Figures 3.1 and 3.3) but a more careful analysis reveals that this is not the case (see Figures 3.2 and 3.4). It is perhaps reasonable to assume that the peaks shown in Figures 3.2 and 3.4 are due to numerical errors but we must nevertheless account for the peaks in the optimization process.

The analysis primarily raises awareness of two facts. First, the problem is not convex and so convex optimization tech-
niques cannot be readily used. Second, we are offered important clues as to how an efficient algorithm may be constructed. Namely, the effect on the residuals is greater by orders of magnitude as we go along $I_\chi$ rather than $T_N$. It seems reasonable, therefore, that stepping along $I_\chi$ primarily in the first iterations of the algorithm should offer greater efficiency than if we step along $T_N$.

It should be clear that $\delta$-almost convexity can be applied along $I_\chi$. However, the variation in the residuals over $T_N$ is not as marked so we may have to evaluate all values (up to some subdivision) within the bounds on $T_N$.

We shall remark that due to the “flat” nature of the residuals in $T_N$, it is difficult to find the $T_N$-optimal point. It is therefore necessary to investigate further how to account for this problem in an efficient manner.
Figure 3.1: At first inspection, \( \mathcal{R} \) appears to be convex.
Figure 3.2: $\mathcal{R}$ is seen to be non-convex when viewed closely.
3.4. RESIDUAL ANALYSIS

Figure 3.3: At first inspection, $\mathcal{R}$ appears to be convex.
Figure 3.4: $\mathcal{R}$ is seen to be non-convex when viewed closely.
3.5 Proposed Fitting Algorithm

In the very first step we center our data using the offsetting process described earlier. We then remove all current-points for which the current is \( I_b < 0 \) as the current-voltage model is only defined for \( I_b \geq 0 \).

Let \( \theta = (I_C, R_N, T_N, I_X) \), \( v_X(I_b; \theta) \) be the extended current-voltage model described earlier and \( v_{RSJX}(I_b; \theta) \) be the RSJ model shifted by \( I_X \). We then define

\[
v(I_b; \theta) = \begin{cases} 
  v_X(I_b; \theta) & 0 \leq I_b \leq \xi_\theta, \\
  v_{RSJX}(I_b; \theta) & \xi_\theta < I_b < \infty,
\end{cases}
\]  

(3.19)

where \( \xi_\theta \) is determined for each \( \theta \) so that

\[
\left| \frac{v_{RSJX}(\xi_\theta; \theta) - v_X(\xi_\theta; \theta)}{v_X(\xi_\theta; \theta)} \right| \leq f
\]  

(3.20)

for some \( f \in [0, 1] \). (This is fit function switchover.)

Let \( \Lambda = [I_{Cmin}, I_{Cmax}] \times [R_{Nmin}, R_{Nmax}] \) be the parameter space for \( \lambda = (I_C, R_N) \) and \( \Theta = [I_{Xmin}, I_{Xmax}] \times [T_{Nmin}, T_{Nmax}] \) be the parameter space for \( \theta = (I_X, T_N) \). Finding \( \Lambda \) can be done by stepping in current to find a threshold voltage and establishing bounds on \( I_C \) based on the current-point at which the threshold voltage is exceeded. For \( R_N \), a linear regression near the endpoint of the current data will yield reasonable bounds based on the normal-state assumption discussed earlier. Initial guesses for \( I_C \) and \( R_N \) are obtained simply by registering the \( I_C \)-point found and the slope of the regression line, respectively.
For Λ, we use stock convex fitting algorithms.¹ For Θ, we describe the fitting process below. Each residual described is obtained from fixing θ ∈ Θ and using the convex algorithm.

We provide some δ > 0 for which it is reasonable the residual function R over T_N and I_ξ is δ-almost convex. For a fixed T_N, we then step along I_ξ, using a step size h, starting from I_ξ = 0, until we have reached a point at which R - R_{min} > δ, where R_{min} is the minimal value of R so far achieved. At this point we need not step any further in I_ξ as we are guaranteed to have reached the minimal point by convexity. We then proceed by stepping backwards with a smaller step size h' < h until we reach either I_ξ = 0, or R - R_{min} > δ.

We then use the minimal-residual point I^*_ξ and start stepping in T_N. Here, we simply step over an interval of T_N-values using a step size h'' that is unrelated to h and h'. The minimal-residual point (I^*_ξ, T^*_N) corresponds to our (approximately) optimal point.

A (partial) implementation of the algorithm is given in Appendix C

### 3.6 Parallelization

The algorithm lends itself well to parallelization. We have chosen to apply parallel processing in the evaluation of the fitting function as the current (I_ξ) points are independent. However,

¹This has experimental support but needs to be verified further as a future research direction.
to reap the benefits from the fit function switchover process described earlier, we must divide $I_b$ into chunks that we evaluate in order.

Let $(X, Y)$ represent our data, where $X$ is the set of $I_b$ values and $Y$ is the set of voltages. We define a chunk size $S_C$, normally kept at or below the number of processing cores available, $N_P$. Denote by $\#X$ the number of elements in $X$. We then partition $X$ into $N_C := \#X / S_C$ sets $X_1, \ldots, X_{N_C}$ such that for all $x \in X_i$ and $x' \in X_j$, we have $x < x'$ whenever $i < j$.

We then evaluate the current model, in order, on $X_1, \ldots, X_{N_C}$. Whenever the fit function is within the prescribed bounds of the approximate fit function (as described earlier), we switch the following chunks to evaluate only the approximate function.
Appendices
Appendix A

Implantation Studies

The following is an excerpt from a report [35] submitted for the UC Scholars Program 2014 during which the present author conducted fabrication studies of Josephson devices. The work was supervised by Shane A. Cybart, Ethan Y. Cho, Travis J. Wong and Robert C. Dynes and was supported by the UC Scholars Program and the Air Force Office of Scientific Research (AFOSR).

One method of fabricating Josephson junctions is through an ion irradiation process. In this process, ions are accelerated using an electric field and the ions then collide with a superconducting material. This results in atoms in the superconductor getting displaced from their original lattice sites. It has been shown in numerous studies, for example [37] and [33], that $T_C$ can be controlled through this process.

Therefore, if one wishes to create a Josephson junction with
Figure A.1: Effect on transition temperature in YBCO after ion irradiation as a function of fluence. It can be seen that the material transitions from a superconductor to a normal metal to an insulator with increasing fluence. (Adapted from Ref. [20])

an operating temperature of $T$, a superconductor with transition temperature $T_C > T$ can be irradiated in a barrier region so that this region has a transition temperature $T_C' < T$. (See Figure A.1.)

In our experiments, we use an epitaxially grown 30 nm YBCO film on a sapphire substrate. A common operating temperature is 77 K, which is the boiling point of nitrogen and which is compatible with YBCO, with $T_C$ around 90 K [38].
A.1 Junction Geometries

Our main interest in this work lies in designing junctions with varying geometries. We have established a baseline set of junctions in 30 nm YBCO film, but it is desirable to increase the film thickness. This is because increasing the thickness will allow downsizing other dimensions, thus allowing for a higher density of junctions on a single chip. Herein, we have simulated 250 nm YBCO films which will allow scaling down junction width to 12% of the current width while maintaining junction surface area.

A.2 Prior Work

Ion irradiation as a means for fabrication of Josephson junctions has been extensively studied. Clark et. al [10] quantify the amorphization and $T_C$, etc., effects that are incurred upon implanting oxygen and arsenic ions into YBCO. These results are extended in [9], where emphasis is on the crystal microstructure after implantation in the material. White et. al [36] demonstrate that nuclear collisions are the primary determinator of the damage profile in ion-irradiated YBCO and use data to show that ion irradiation destroys superconducting coherence. White et. al [37] show that Josephson critical current can be controlled through ion beam irradiation. Valles et. al [33] show that YBCO resistivity after ion bombardment is a function of carrier mobility rather than carrier density and quantify $T_C$ as a function of ion dose. Summers et. al [30] show a directly proportional relationship between $T_C$ and damage induced through ion im-
plantation in YBCO. Lang et. al shows YBCO transition temperature decrease as a function of ion dose [20]. Cybart et. al [13] compare the uniformity of YBCO junctions after irradiation by helium and neon ions.

Studies on Monte-Carlo methods for simulation has also been carried out, see for example Peng et. al [26], which compares YBCO damage profiles as a function of ion species, using said methods.

A.3 Results

There are several factors to consider when choosing a dose-energy pair. First, a high enough energy is required to penetrate the material fully. However, as energy increases, so must dose or the density of damage will not be enough to create a barrier. Figure A.4 showcases these effects. Also important is lateral straggle, which is the spread of damage normal to the direction of irradiation. More lateral straggle means that the barrier becomes thicker, which has negative effects on junction parameters.

Currently, it is infeasible to simulate every single ion trajectory in the irradiation process, so we therefore use pseudo-ions. One pseudo-ion represents several real ions, so we can use the number of pseudo-ions to simulate as a parameter that trades off accuracy for simulation time. Figure A.3 showcases this effect. We started using $75 \times 10^3$ pseudo-ions as these simulations complete relatively quickly while giving at least high-level information about the damage profile. For more in-depth informa-
Figure A.2: Simulations of varying dose-energy pairs for a 250 nm thick YBCO film. a) $10 \times 10^{16}$ ions, 10 keV. This energy is too low to fully penetrate the material. Also note the large lateral straggle. b) $10 \times 10^{16}$ ions, 55 keV is enough to fully penetrate the material but some lateral straggle still remains. c) $7 \times 10^{16}$ ions, 150 keV. In this case, the dose is too low and therefore the density of damage is low. d) $19 \times 10^{16}$ ions, 150 keV. Here, a relatively high dose and energy creates a uniform thin barrier which is likely to give rise to a functional junction.
tion, we have run mostly $1 \times 10^6$ pseudo-ions, but have gone as high as $2 \times 10^6$ pseudo-ions.

From observing lateral straggle on a contour plot, we selected 100 keV as a candidate energy for 250 nm YBCO films. The quantity $Y/Y_{\text{max}}$ was introduced and is the distance from the film surface into the film, normalized to lie in the interval $[0, 1]$. Using this quantity, we may compare damage profiles across different film thicknesses. Using 30 nm thick junctions known to be operational at $\sim 15$ K and $\sim 77$ K we ran simulations of the dose/energy parameters we used to fabricate those junctions. Plotting $Y/Y_{\text{max}}$ vs damage (in displacements per cubic centimeter) for both the 30 nm and 250 nm films, we may the compare the damage profiles. In accordance with [30] and [13], we see a linearly proportional change in $T_C$ with ion dose. Furthermore, we see a damage profile similar to that of the $\sim 15$ K junction with $9 \times 10^{16}$ ions/cm$^2$ and a profile similar to that of the $\sim 77$ K junction with $24 \times 10^{16}$ ions/cm$^2$.

\section*{A.4 Future Work}

Future work includes quantifying lateral straggle, developing automated fitting routines for damage profiles and correlating $T_C$ with damage density. When these are completed, a correlation between dose/energy and maximum junction thickness can be made.

Because the method is not specific to YBCO, other materials such as MgB$_2$ are candidates for the method, too. To facilitate processing, new software has to be developed that can commis-
A.4. FUTURE WORK

sion and control computations.
Figure A.3: Simulation of irradiation of a 30 nm thick YBCO film using 30 keV He$^+$ ions and with $75 \times 10^3$, $150 \times 10^3$, and $10 \times 10^6$ pseudo-ions representing $2 \times 10^{16}$ real ions/cm$^2$. Here $Y/Y_{\text{max}}$ represents the depth into the film with 0 being the surface and 1 being 30 nm. As can be seen from the graph, a higher number of pseudo-ions yields a more uniform damage profile.
Figure A.4: Simulations of 100 keV He$^+$ at 9, 14, 19, $20 \times 10^{16}$ ions/cm$^2$ irradiating 250 nm YBCO. The graph shows damage density along the mid-line of the barrier region. $Y/Y_{max}$ is normalized depth. Simulated damage curves for 30 nm junctions known to be operational at $\sim 15$ K and $\sim 77$ K are shown in red for reference.
Appendix B

Code: Residual Analysis

```matlab
function [lx, Tn, Res] = GetResidualsSurface(dataset, T)

tic

IX_DEVS = 0.5;

if nargin < 2
  warning('No temperature provided! Using default value T=60 K...');
  T = 60;
end

dataset = sortrows(dataset, 1);
dataset = StripLeft(dataset);

[guess_Rn, guess_lx] = GuessRnIx(dataset);
guess_lc_base = GuessIc(dataset);

LB_Rn = guess_Rn * (1 - 0.05);
UB_Rn = guess_Rn * (1 + 0.03);

range_low_lx = ClampLow(guess_lx - IX_DEVS * guess_lx, 0);
range_high_lx = guess_lx + IX_DEVS * guess_lx;

ival_lx = transpose(linspace(0, range_high_lx, 64));
disp(ival_lx)

ival_Tn = transpose(65:3:120);

[grid_lx, grid_Tn] = meshgrid(ival_lx, ival_Tn);
```
APPENDIX B. CODE: RESIDUAL ANALYSIS

ParStart(32);

num_Ix = size(ival_Ix, 1);
num_Tn = size(ival_Tn, 1);

residuals_flattened = zeros(num_Ix * num_Tn, 1);
parfor k=1:num_Ix * num_Tn
  i = ceil(k / num_Ix);
  j = mod(k - 1, num_Ix) + 1;
  cur_Ix = grid_Ix(i, j);
  cur_Tn = grid_Tn(i, j);
  fprintf('Fitting with Ix: %d, Tn: %d, ', cur_Ix, cur_Tn);
  if guess_Ic < LB_Ic || guess_Ic > UB_Ic
    warning('Ic guess outside bounds, defaulting to midpoint...');
    guess_Ic = (LB_Ic + UB_Ic) / 2;
  end
  fprintf('Ic: [ %d< ( %d ) <%d ]', LB_Ic, guess_Ic, UB_Ic);
  fprintf('
');
  LB = [LB_Ic, LB_Rn];
  UB = [UB_Ic, UB_Rn];
  guesses = [guess_Ic, guess_Rn];
  residuals_flattened(k) = Fit(dataset, cur_Ix, cur_Tn, guesses, ... LB, UB);
end

residuals = zeros(num_Tn, num_Ix);
for k=1:num_Ix * num_Tn
  i = ceil(k / num_Ix);
  j = mod(k - 1, num_Ix) + 1;
  residuals(i, j) = residuals_flattened(k);
end

dlmwrite('output/grid_Ix.txt', grid_Ix);

dlmwrite('output/grid_Tn.txt', grid_Tn);

dlmwrite('output/residuals.txt', residuals);
total_time = toc;

dlmwrite('output/runtime.txt', [total_time]);
end

function resnorm = Fit(dataset, Ix, Tn, guesses, LB, UB)

options = optimset(...
    'MaxFunEvals', 400, ...
    'TolFun', 1e-10, ...
    'TolX', 1e-10, ...
    'MaxIter', 600, ...
    'Display', 'Off');

fitfunction = @(pars, x) IVModel(x, pars(1), pars(2), Ix, Tn);

[fit, resnorm] = lsqcurvefit(fitfunction, ...
    guesses, ...
    dataset(:, 1), dataset(:, 2), ...
    LB, UB, ...
    options);

filename = ['output/Ic_', num2str(fit(1)*1E6), '_Rn_', ...
    num2str(fit(2)), '_Ix_', num2str(Ix), '_Tn_', num2str(Tn), ...
    '_R_', num2str(resnorm)];

X = linspace(dataset(1, 1), dataset(end, 1), 100);
Y = IVModel(X, fit(1), fit(2), Ix, Tn);

dlmwrite([filename 'X.txt'], X);
dlmwrite([filename 'Y.txt'], Y);
end

function val_out = ClampLow(val_in, clamp_val)
    val_out = val_in;
    if val_in < clamp_val
        val_out = clamp_val;
    end
end

function val_out = ClampHigh(val_in, clamp_val)
    val_out = val_in;
    if val_in > clamp_val
        val_out = clamp_val;
    end
end

% Gets the index of the I-V point where I is first >= 0
function midPtIdx = GetDataMidPtIdx(dataset)
    midPtIdx = 0;
    for i=1:size(dataset, 1)
        if dataset(i, 1) >= 0
            midPtIdx = i;
            break;
        end
    end
    if midPtIdx == 0
        error('Could not find midpoint...');
    end
end

% Returns the right part of the dataset
function dataset = StripLeft(dataset)
    midPtIdx = GetDataMidPtIdx(dataset);
    dataset = dataset(midPtIdx:end, :);
end

function [guess_Rn, guess_lx] = GuessRnIx(dataset)
    rangeStart = ceil(size(dataset, 1) * 0.9);
    p = polyfit(dataset(rangeStart:end, 1), dataset(rangeStart:end, 2), 1);
    guess_Rn = p(1);
    guess_lx = -p(2)/p(1);
end

function guess_Ic = GuessIc(dataset)
    max_V = max(dataset(:, 2));
    threshold_V = max_V * 0.25;
    guess_Ic = 1E-3;
    for i=1:size(dataset, 1)
        if dataset(i, 2) >= threshold_V
            guess_Ic = dataset(i, 1);
            break;
        end
    end
end
function V = IVModel(Ib, Ic, Rn, Ix, Tn)

hbar = 4.135667516E-15 / (2 * pi);
echarge = 1.602176655E-19;
kboltz = 8.6173324E-5;

ah_gamma = hbar * Ic / (echarge * kboltz * Tn);

if size(Ib, 1) < size(Ib, 2)
    Ib = transpose(Ib);
end

V = zeros(size(Ib, 1), 1);

chi = Ix / Ic;
% Fit function switchover
switch_over_reached = 0;

for i=1:size(Ib, 1)
    xv = Ib(i, 1) / Ic;
    if (xv - chi) < 0
        V(i, 1) = 0;
    else
        V(i, 1) = sqrt((xv - chi).^2);
        if switch_over_reached == 0
            U = @(theta) ((1/2) * Tn * ah_gamma) * ((xv - chi) ... *
                .^ theta + cos(theta));
            f = @(theta) exp(-U(theta) ./ Tn);
            if (abs(f(2*pi)) ~= inf & & abs(f(-2*pi)) ~= inf)
                vv = @(x) (4 * pi / ah_gamma) * ( ... *
                    exp(pi * ah_gamma * (x-chi) - 1).^(-1) ... *
                    quad2d(@(th, thp) f(th)/f(thp), ... *
                        0, 2*pi, 0, @(y) y) ... *
                    + quad2d(@(th, thp) f(th)/f(thp), ... *
                        0, 2*pi, @(y) y, 2*pi) ... *
                    .^(-1));
                ah_V = vv(xv);
            end
        end
    end
end

V(i, 1) = ah_V;
APPENDIX B. CODE: RESIDUAL ANALYSIS

244        end
245    end
246    end
247
248    V(i, 1) = Ic * Rn * V(i, 1);
249
250    end
251 end
252
253    \% Sets up parallel processing
254    function ParStart(cores)
255
256        if size(gcp('nocreate'),1) == 0
257
258            pcluster = parcluster('local');
259            pcluster.NumWorkers = cores;
260            ppool = parpool(pcluster, cores - 1)
261
262        end
263 end
Appendix C

Code: Fitting Implementation

1 function TnSet = FittingAlgo(dataset, T)
2 delete('output/*.txt');
3 if nargin < 2
4     warning('No temperature provided! Using default value T=60 K...');
5     T = 60;
6 end
7 IX_DEVS = 1;
8 tic
9 dataset = sortrows(dataset, 1);
10 dataset = StripLeft(dataset);
11 dlmwrite('output/exdata.txt', dataset);
12 dataset = ThinData(dataset, 10);
13 [guess_Rn, guess_Ix] = GuessRnIx(dataset);
14 guess_Ic = GuessIc(dataset) * 0.3;
15 LB_Rn = guess_Rn * (1 - 0.05);
APPENDIX C. CODE: FITTING IMPLEMENTATION

UB_Rn = guess_Rn * (1 + 0.03);

LB_Ic = ClampLow(0, 0);
UB_Ic = ClampHigh(guess_Ic + guess_Ic * 3, max(dataset(:, 1)));

range_low_Ix = ClampLow(guess_Ix - IX_DEVS * guess_Ix, 0);
range_high_Ix = guess_Ix + IX_DEVS * guess_Ix;

ival_Ix = transpose(linspace(0, range_high_Ix, 100));

%delta almost convexity with delta = 1E-6;
delta_acvx = 5E-7;

num_cpus = 32;
ParStart(num_cpus);

init_Tn = T * 1.1;

IX_residuals = zeros(size(ival_Ix, 1), 2);
residual_lowest = -1;
residual_lowest_Ix = 0;

best_fit = zeros(1,4);

for pass = 1:1
  for cur_Ix_idx = 1:size(ival_Ix, 1)
    fprintf('Ix: %f uA', ival_Ix(cur_Ix_idx, 1)/1E-6);
    guesses = [guess_Ic, guess_Rn];
    LB = [LB_Ic, LB_Rn];
    UB = [UB_Ic, UB_Rn];
    Ix = ival_Ix(cur_Ix_idx, 1);
    [residual, fIx, fTn, fIc, fRn] = Fit(dataset, Ix, init_Tn, ...
      guesses, LB, UB);
    IX_residuals(cur_Ix_idx, :) = [Ix/1E-6, residual];
    if residual_lowest == -1 || residual_lowest > residual
      residual_lowest = residual;
      residual_lowest_Ix = Ix;
      best_fit = [fIx, fTn, fIc, fRn];
    end
    fprintf('Res: %d\n', residual);
    if residual > residual_lowest + delta_acvx
      disp('delta almost convex point reached...');
      break;
    end
  end
end
```plaintext
dlmwrite(['outputf/residuals_lx_p' num2str(pass) '.txt'], ... 
    lx_residuals);
end

Tn_low = T * 0.9;
Tn_high = T * 1.4;
Tn_pts = 100;

Tn_range = transpose(linspace(Tn_low, Tn_high, Tn_pts));
residual_lowest = -1;
best_lclx = best_fit(3) + best_fit(1);
lclx_threshold = 0.3;

TnSet = zeros(0, size(dataset,2));
for i=1:size(dataset,1)
    if dataset(i,1) < best_lclx * (1 + 1.2 * lclx_threshold) ... 
        && dataset (i,1) > best_lclx * (1 - lclx_threshold)
        TnSet(end+1,:) = dataset(i,:);
    end
end

scatter(TnSet(:,1), TnSet(:,2))
for i=1:Tn_pts
    fprintf(' Tn: %d \n', Tn_range(i,1));
    Tn = Tn_range(i,1);
    residual = TnEval(TnSet, best_fit, Tn);
    fprintf(' Res: %f \n', residual);
    if residual_lowest == -1 || residual < residual_lowest
        best_fit = [flx, Tn, flc, fRn]
        residual_lowest = residual;
    end
end
fprintf('BEST FIT: \n
ic: %f uA\n Rn: %f Ohm\n Tn: %f K\n lx: %f uA\n', ... 
    best_fit(3)/1E-6, best_fit(4), best_fit(2), best_fit(1)/1E-6);
fprintf(' Residual: %d\n', residual_lowest);
total_time = toc;
fprintf(''' Total running time: %d min\n', total_time / 60);
dlmwrite('outputf/runtime.txt', [total_time]);
```

function residual = TnEval(dataset, best_fit, Tn)
    Ix = best_fit(1);
    Ic = best_fit(3);
    Rn = best_fit(4);

    V = IVModel(dataset(:,1), Ic, Rn, Ix, Tn);

    residual = sqrt(norm(V - dataset(:,2), 2));
end

function chunks = CreateChunks(eval_points, chunk_size)
    num_points = size(eval_points, 1);
    num_chunks = ceil(num_points / chunk_size);

    chunks = cell(num_chunks, 1);

    for i = 1:num_chunks
        chunk_start_idx = (i - 1) * chunk_size + 1;
        chunk_end_idx = chunk_start_idx + chunk_size - 1;

        if i == num_chunks
            chunk_end_idx = num_points;
        end

        chunks{i, 1} = eval_points(chunk_start_idx:chunk_end_idx, 1);
    end
end

function [resnorm, Ix, Tn, Ic, Rn] = Fit(dataset, Ix, Tn, guesses, ...
    LB, UB)

    options = optimset( ...'
        'MaxFunEvals', 30, ...'
        'TolFun', 1e-11, ...'
        'TolX', 1, ...'
        'MaxIter', 600, ...'
        'Display', 'Off');

    fitfunction = @(pars, x) IVModel(x, pars(1), pars(2), Ix, Tn);

    [fit, resnorm] = lsqcurvefit(fitfunction, ...
        guesses, ...
        dataset(:, 1), dataset(:, 2), ...
        LB, UB, ...
        options);
lc = fit(1);
Rn = fit(2);
filename = [\'output/R_{n}\' num2str(resnorm) \_lc\_ ... 
num2str(fit(1)+1E6) \_Rn\_ ... 
num2str(fit(2)) \_Ix\_ num2str(Ix+1E6) \_Tn\_ num2str(Tn) ... 
\_tx\_];
X = linspace(dataset(1, 1), dataset(end, 1), 100);
Y = IVModel(X, fit(1), fit(2), Ix, Tn);
dlmwrite(filename, [X', Y]);
%dlmwrite([filename ' X.txt'], X);
%dlmwrite([filename ' Y.txt'], Y);

function dataset_thinned = ThinData(dataset, point_spacing)
dataset_thinned = zeros(0, size(dataset, 2));
for i=1:point_spacing:size(dataset, 1)
dataset_thinned(end+1,:) = dataset(i,:);
end
end

function val_out = ClampLow(val_in, clamp_val)
val_out = val_in;
if val_in < clamp_val
val_out = clamp_val;
end
end

function val_out = ClampHigh(val_in, clamp_val)
val_out = val_in;
if val_in > clamp_val
val_out = clamp_val;
end
end

% Gets the index of the I–V point where I is first >= 0
function midPtIdx = GetDataMidPtIdx(dataset)
midPtIdx = 0;
for i=1:size(dataset, 1)
if dataset(i, 1) >= 0
midPtIdx = i;
break;
end
end
end
APPENDIX C. CODE: FITTING IMPLEMENTATION

```matlab
if midPtIdx == 0
    error('Could not find midpoint...');
end

% Returns the right part of the dataset
function dataset = StripLeft(dataset)
    midPtIdx = GetDataMidPtIdx (dataset);
    dataset = dataset(midPtIdx:end,:);
end

definition [guess_Rn, guess_Ix] = GuessRnIx(dataset)
    rangeStart = ceil(size(dataset, 1) * 0.9);
    p = polyfit(dataset(rangeStart:end, 1), ...
                dataset(rangeStart:end, 2), 1);
    guess_Rn = p(1);
    guess_Ix = -p(2)/p(1);
end

definition guess_Ic = GuessIc(dataset)
    max_V = max(dataset(:, 2));
    threshold_V = max_V * 0.25;
    guess_Ic = 1E-3;
    for i=1:size(dataset, 1)
        if dataset(i, 2) >= threshold_V
            guess_Ic = dataset(i, 1);
            break;
        end
    end
end

definition V=IVModel(Ib, Ic, Rn, Ix, Tn)
    fprintf('lc: %f, Rn: %f, Ic/I/6, Rn');
    hbar = 4.135667516E-15 / (2 * pi);
    echarge = 1.602176565E-19;
    kboltz = 8.6173324E-5;
    a_hgamma = hbar * Ic / (echarge * kboltz * Tn);
```
if size(Ib, 1) < size(Ib, 2)
    lb = transpose(lb);
end

V = zeros(size(lb, 1), 1);

chi = lx / 1c;

U = @(theta, x) (-1/2) * Tn * ah_gamma) * ((x-chi) ...
    .* theta + cos(theta));

f = @(theta, x) exp(-U(theta, x) ./ Tn);

vv = @(x) (4 * pi / ah_gamma) * ( ...
    exp(pi * ah_gamma * (x-chi)) - 1).^(-1) ...
    * quad2d(@(th, thp) f(th, x)./f(thp, x), 0, 2*pi, ...
    0, @(y) y) ...
    + quad2d(@(th, thp) f(th, x)./f(thp, x), 0, 2*pi, ...
    @(y) y, 2*pi) ...
    ).^(-1);

% Break problem into chunks for parallel processing

chunks_Ib = CreateChunks(lb, 31);
chunks_V = cell(size(chunks_Ib));

switch_over_reached = 0;

for cur_chunk=1:size(chunks_Ib, 1)
    chunk_Ib = chunks_Ib{cur_chunk, 1};
    chunk_size = size(chunk_Ib, 1);

    chunk_V = zeros(chunk_size, 1);
    pct_diff = zeros(chunk_size, 1);

    if switch_over_reached == 0
        for i=1:chunk_size
            xval = chunk_Ib(i, 1) / 1c;

            if (xval - chi) < 0
                chunk_V(i, 1) = 0;
            else
                rsj_V = sqrt((xval - chi).^2-1);

                if (abs(f(2*pi, xval)) == inf && abs(f(-2*pi, xval)) == inf)
                    ah_V = vv(xval);

                    pct_diff(i, 1) = abs((rsj_V - ah_V) / ah_V);
                end

                chunk_V(i, 1) = ah_V;
            end
        end
    end % parfor
APPENDIX C. CODE: FITTING IMPLEMENTATION

if max(pct_diff) < 0.02
    switch_over_reached = 1;
    fprintf('%SO<%d/%d>', cur_chunk, size(chunks_lbd, 1));
else
    parfor i = 1:chunk_size
        xval = chunk_lbd(i, 1) / Ic;
        if (xval - chi) < 0
            chunk_V(i, 1) = 0;
        else
            chunk_V(i, 1) = sqrt((xval - chi)^2 - 1);
        end
    end
end

chunks_V{cur_chunk, 1} = Ic * Rn * chunk_V;
end

end

function ParStart(cores)
    if size(gcp('nocreate'), 1) == 0
        pcluster = parcluster('local');
        pcluster.NumWorkers = cores;
        ppool = parpool(pcluster, cores - 1)
    end
end
Bibliography


BIBLIOGRAPHY


