# Defect-induced Flat Bands in Topological Superconductors and Semimetals

UBC Science Coop
Max Planck Institute for Solid State Research

**Student:** Mohamed Shaaban
**Supervisor:** Dr. Andreas Schnyder

February 11, 2019

# Contents

# 1 Introduction

In recent years topological materials have gained the attention of researchers as a result of exhibiting new physical phenomenon with promising applications(e.g. quantum information technology). One of these phenomena is the ability to support gapless states localized on the edge(boundary) or on defects. This phenomenon is known as the bulk-boundary(defect) correspondence[1, 2, 5]. Topological states can be realized in a variety of systems, the systems relevant for this report are noncentrosymmetric and nodal superconductors.

The gapless boundary modes resultant from bulk-boundary(defect) correspondence can manifest as flat-bands(highly degenerate states)[1]. Such zero energy flat-bands are of interest as they have a macroscopically degenerate ground state and provide exotic new ground states in the presence of interactions.

## 1.1 Project Description

My project is on studying flatbands induced as a result of bulk-defect correspondence in topological superconductors and semimetals. The models studied in this report are the noncentrosymmetric and f-wave superconductors. The main goal is to find if zero energy flat-bands are induced by and localized on a variety of defects. The defects investigated in this report are vortices, dislocations and strain.

The models are given as k-space Hamiltonians, they are then Fourier-transformed to real space in order to implement the defects as spatially dependant variations in the Hamiltonian. The Hamiltonian is then diagonalized to find the eigenvalues and eigenvectors and he energy spectrums are then examined to observe the flat-bands. The implementations are all done in MATLAB, sample programs are provided in the appendix.

## 2 Vortex Implementation

The vortex implementation is directly based on the work of a previous intern, the original work can be found here[3]. Below is reiteration of the original work with a modification on the physical location of the vortex implementation. Note that all models examined with a vortex are 2D square lattices.

In order to enable implementation of periodic boundary conditions the vorticies have to be paired with anitvortices. Let the system be an N-by-N integer lattice where $x, y \in [-N/2, N/2)$. Note that the vortex/antivortex pair must be implemented between sites to avoid the rise of a singularity. As a result the pair is implemented at $(x, y) = (\pm a, 0)$ on a (+1/2,+1/2) shifted N-by-N lattice(see figure 1). We choose $a = N/4$ to maximize the distance between the vortex and antivortex.

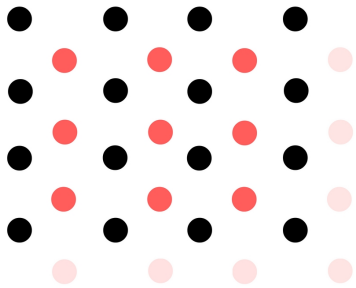


Figure 1: N-by-N real space lattice(Black) with N-by-N half shifted vortex lattice(Red)

The actual implementation of the vortices is done by implementing the following spatial dependency on the gap parameter

$$\Delta_0 \to \Delta_0 \tanh(r_a/c)\tanh(r_v/c)e^{i\omega\phi} \tag{1}$$

where $r_v, r_a$ are the distances from the vortex and antivortex respectively. The parameter c detrmines the vortex sizw and the parameter $\omega$ is the vorticity. For the sake of simplicity they will both be taken as 1 in this report. $\phi(x, y)$ is a spatially dependant parameter defined by the following equation

$$\phi(x, y) = \arctan\left(\left(\frac{2ay}{x^2 + y^2 - a^2}\right)\left(1 - \frac{|y|}{N/2} + \alpha\right)^{1/\beta}\right) \tag{2}$$

where $\alpha$ and $\beta$ are variables to be optimized to ensure periodic continuity. For more on the optimization see the original implementation [3].

# 3 Dislocation Implementation

The dislocations implemented in this report are all along the x direction of a 2D lattice(or a 3D lattice composed of 2D layers). While the choice of direction is arbitrary the implementation must be direction specific to ensure the periodicity implementation is consistent with the dislocation.

The dislocation is implemented by constructing two separate systems, an N-by-N integer real space lattice(system b) and an (N-1)-by-(N-1) real space lattice(system d). In order to apply periodic boundary conditions the system must be mirrored resulting in a total of 4 systems. The 4 systems are aligned along the x direction(Consult figure2) which is represented by a block matrix of the form:

$$\begin{pmatrix} dl & 0 & 0 & 0 \\ 0 & bl & 0 & 0 \\ 0 & 0 & br & 0 \\ 0 & 0 & 0 & dr \end{pmatrix} \tag{3}$$

where $dl, bl, br, dr$ represent d-left, b-left, b-right, d-right respectively.

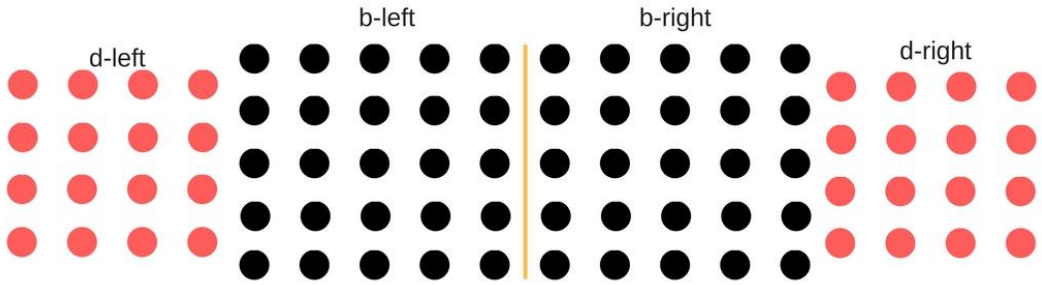



Figure 2: 4 systems aligned along the x direction

The systems are then coupled by adding the appropriate hopping terms. We let y run from 1 to (N-1), if $y \leq (N-1)/2$ then we couple the systems by:

$$((N-1)_{dl}, y) \leftrightarrow (1_{bl}, y), \quad (N_{br}, y) \leftrightarrow (1_{dr}, y) \quad \text{hopping} \tag{4}$$

If $y > (N-1)/2$ then we couple the systems by:

$$((N-1)_{dl}, y) \leftrightarrow (1_{bl}, y+1), \quad (N_{br}, y+1) \leftrightarrow (1_{dr}, y) \quad \text{hopping} \tag{5}$$

This implementation requires N be an odd integer. We note that the system can be arranged such that it dislocates inwards rather than outwards (dl and dr are placed in the center), however no difference in the end result since Hamiltonians are Hermitian.

# 4 Strain Implementation

TO BE COMPLETED(See Further Work (sec 7))

# 5 Noncentrosymmetric Superconductor

## 5.1 Definition of Hamiltonian

The $\boldsymbol{k}$-space Hamiltonian if defined as follows $\mathcal{H}(\boldsymbol{k}) = \sum_{\boldsymbol{k}} \psi^{\dagger}_{\boldsymbol{k}} H(\boldsymbol{k}) \psi_{\boldsymbol{k}}$ with $\psi_{\boldsymbol{k}} = (c_{\boldsymbol{k}\uparrow}, c_{\boldsymbol{k}\downarrow}, c^{\dagger}_{-\boldsymbol{k}\uparrow}, c^{\dagger}_{-\boldsymbol{k}\downarrow})^T$ where $c_{\boldsymbol{k}\sigma}$ is the electron annihilation operator with spin $\sigma$ and momentum $\boldsymbol{k}$. [6]

$$H(\boldsymbol{k}) = \begin{pmatrix} h(\boldsymbol{k}) & \Delta(\boldsymbol{k}) \\ \Delta^{\dagger}(\boldsymbol{k}) & -h^T(-\boldsymbol{k}) \end{pmatrix} \tag{6}$$

where

$$h(\boldsymbol{k}) = \epsilon_{\boldsymbol{k}} \sigma_0 + \boldsymbol{g}_{\boldsymbol{k}} \cdot \boldsymbol{\sigma} \tag{7}$$

$$\Delta(\boldsymbol{k}) = (\Delta_s \mathbf{1} + \boldsymbol{d}_{\boldsymbol{k}} \cdot \boldsymbol{s})(i s_y) \tag{8}$$

and

$$\epsilon_{\boldsymbol{k}} = t(cos(k_x) + cos(k_y)) - \mu \tag{9}$$

$$\boldsymbol{g}_{\boldsymbol{k}} = \alpha \boldsymbol{l}_{\boldsymbol{k}} \tag{10}$$

$$\boldsymbol{d}_{\boldsymbol{k}} = \Delta_p \boldsymbol{l}_{\boldsymbol{k}} \tag{11}$$

In these definitions, $\boldsymbol{\sigma}$ and $\boldsymbol{s}$ represent vectors of Pauli matrices, and $\boldsymbol{l}_{\boldsymbol{k}}$ is the order parameter vector. The order parameter depends on the point group symmetry. below is a table of point group symmetries from pg 2390 in [4].

| $\mathbb{G}$ | $\gamma(\boldsymbol{k})$ | Zeros of $\gamma(\boldsymbol{k})$ |
|---|---|---|
| $\mathbf{C}_1$ (1) | $(a_1 k_x + a_2 k_y + a_3 k_z)\hat{x} + (a_4 k_x + a_5 k_y + a_6 k_z)\hat{y} + (a_7 k_x + a_8 k_y + a_9 k_z)\hat{z}$ | point |
| $\mathbf{C}_2$ (2) | $(a_1 k_x + a_2 k_y)\hat{x} + (a_3 k_x + a_4 k_y)\hat{y} + a_5 k_z \hat{z}$ | point |
| $\mathbf{C}_s(m)$ | $a_1 k_z \hat{x} + a_2 k_z \hat{y} + (a_3 k_x + a_4 k_y)\hat{z}$ | point |
| $\mathbf{D}_2$ (222) | $a_1 k_x \hat{x} + a_2 k_y \hat{y} + a_3 k_z \hat{z}$ | point |
| $\mathbf{C}_{2v}(mm2)$ | $a_1 k_y \hat{x} + a_2 k_x \hat{y} + i a_3 (k_+^2 - k_-^2) k_z \hat{z}$ | line |
| $\mathbf{C}_4$ (4) | $(a_1 k_x + a_2 k_y)\hat{x} + (-a_2 k_x + a_1 k_y)\hat{y} + a_3 k_z \hat{z}$ | point |
| $\mathbf{S}_4(\bar{4})$ | $(a_1 k_x + a_2 k_y)\hat{x} + (a_2 k_x - a_1 k_y)\hat{y} + (b k_+^2 + b^* k_-^2) k_z \hat{z}$ | line |
| $\mathbf{D}_4$ (422) | $a_1 (k_x \hat{x} + k_y \hat{y}) + a_2 k_z \hat{z}$ | point |
| $\mathbf{C}_{4v}(4mm)$ | $a_1 (k_y \hat{x} - k_x \hat{y}) + i a_2 (k_+^4 - k_-^4) k_z \hat{z}$ | line |
| $\mathbf{D}_{2d}(\bar{4}2m)$ | $a_1 (k_x \hat{x} - k_y \hat{y}) + a_2 (k_+^2 + k_-^2) k_z \hat{z}$ | line |
| $\mathbf{C}_3$ (3) | $(a_1 k_x + a_2 k_y)\hat{x} + (-a_2 k_x + a_1 k_y)\hat{y} + a_3 k_z \hat{z}$ | point |
| $\mathbf{D}_3$ (32) | $a_1 (k_x \hat{x} + k_y \hat{y}) + a_2 k_z \hat{z}$ | point |
| $\mathbf{C}_{3v}(3m)$ | $a_1 (k_y \hat{x} - k_x \hat{y}) + a_2 (k_+^3 + k_-^3)\hat{z}$ | line |
| $\mathbf{C}_6$ (6) | $(a_1 k_x + a_2 k_y)\hat{x} + (-a_2 k_x + a_1 k_y)\hat{y} + a_3 k_z \hat{z}$ | point |
| $\mathbf{C}_{3h}(\bar{6})$ | $(b_1 k_+^2 + b_1^* k_-^2) k_z \hat{x} + i(b_1 k_+^2 - b_1^* k_-^2) k_z \hat{y} + (b_2 k_+^3 + b_2^* k_-^3)\hat{z}$ | line |
| $\mathbf{D}_6$ (622) | $a_1 (k_x \hat{x} + k_y \hat{y}) + a_2 k_z \hat{z}$ | point |
| $\mathbf{C}_{6v}(6mm)$ | $a_1 (k_y \hat{x} - k_x \hat{y}) + i a_2 (k_+^6 - k_-^6) k_z \hat{z}$ | line |
| $\mathbf{D}_{3h}(\bar{6}m2)$ | $a_1 [i(k_+^2 - k_-^2) k_z \hat{x} - (k_+^2 + k_-^2) k_z \hat{y}] + i a_2 (k_+^3 - k_-^3)\hat{z}$ | line |
| $\mathbf{T}$ (23) | $a(k_x \hat{x} + k_y \hat{y} + k_z \hat{z})$ | point |
| $\mathbf{O}$ (432) | $a(k_x \hat{x} + k_y \hat{y} + k_z \hat{z})$ | point |
| $\mathbf{T}_d(\bar{4}3m)$ | $a[k_x (k_y^2 - k_z^2)\hat{x} + k_y (k_z^2 - k_x^2)\hat{y} + k_z (k_x^2 - k_y^2)\hat{z}]$ | 3 lines |

## 5.2 Implementation of Hamiltonian

To implement the Hamiltonian it is first converted to real space by taking the Discrete Fourier Transform. The real space Hamiltonian is of size $4N^2 \times 4N^2$ and is implemented as an $N \times N$ matrix of $N \times N$ matrices ($H_i$) of $4 \times 4$ matrices ($h_i$).

We define the real space Hamiltonian as

$$H = \begin{pmatrix} H_0 & H_1 & H_2 & & & H_2^\dagger & H_1^\dagger \\ H_1^\dagger & H_0 & H_1 & \ddots & & & H_2^\dagger \\ H_2^\dagger & H_1^\dagger & H_0 & \ddots & & & \\ & \ddots & \ddots & H_0 & & & \\ & & & & \ddots & & \\ H_2 & & & & & & \\ H_1 & H_2 & & & & & \end{pmatrix} \tag{12}$$

$$H_0 = \begin{pmatrix} h_0 & h_1 & & & & h_1^\dagger \\ h_1^\dagger & h_0 & h_1 & & & \\ & h_1^\dagger & h_0 & \ddots & & \\ & & \ddots & h_0 & & \\ & & & & \ddots & \\ h_1 & & & & & \end{pmatrix} \tag{13}$$

$$H_1 = \begin{pmatrix} h_2 & 0 & h_3 & & h_3^\dagger & 0 \\ 0 & h_2 & 0 & \ddots & & h_3^\dagger \\ h_3^\dagger & 0 & h_2 & \ddots & & \\ & \ddots & \ddots & h_2 & & \\ & & & & \ddots & \\ h_3 & & & & & \\ 0 & h_3 & & & & \end{pmatrix} \tag{14}$$

$$H_2 = \begin{pmatrix} 0 & h_4 & & & h_4^\dagger \\ h_4^\dagger & 0 & h_4 & & \\ & h_4^\dagger & 0 & \ddots & \\ & & \ddots & 0 & \\ & & & & \ddots \\ h_4 & & & & \end{pmatrix} \tag{15}$$

Where $h_0$ is the on-site term, $h_1$ and $h_2$ are the nearest neighbour hopping terms in the x and y direction respectively. $h_3$ and $h_4$ are the 4th nearest hopping terms, describing $(x, y) \to (x + 2, y + 1)$ and $(x, y) \to (x + 1, y + 2)$ respectively. Below I list the values for $h_i$ by point group for the three point groups investigated in this project ($\boldsymbol{C_{4v}}$, $\boldsymbol{O}$ and $\boldsymbol{T_d}$). The Pine Green entrees are only present for periodic BC.

### 5.2.1 $C_{4v}$

For this point group symmetry we use the following order parameter and hopping terms.

$$\boldsymbol{l_k} = sin(k_y)\hat{x} - sin(k_x)\hat{y} \tag{16}$$

$$h_0 = \begin{pmatrix} -\mu & 0 & 0 & \Delta_s \\ 0 & -\mu & -\Delta_s & 0 \\ 0 & -\Delta_s^* & \mu & 0 \\ \Delta_s^* & 0 & 0 & \mu \end{pmatrix} \tag{17}$$

$$h_1 = \begin{pmatrix} t/2 & -\alpha/(2i) & \Delta_p/(2i) & 0 \\ -\alpha/(2i) & t/2 & 0 & -\Delta_p/(2i) \\ \Delta_p^*/(2i) & 0 & -t/2 & -\alpha/(2i) \\ 0 & -\Delta_p^*/(2i) & -\alpha/(2i) & -t/2 \end{pmatrix} \tag{18}$$

$$h_2 = \begin{pmatrix} t/2 & \alpha/2 & -\Delta_p/2 & 0 \\ -\alpha/2 & t/2 & 0 & -\Delta_p/2 \\ \Delta_p^*/2 & 0 & -t/2 & -\alpha/2 \\ 0 & \Delta_p^*/2 & \alpha/2 & -t/2 \end{pmatrix} \tag{19}$$

$$h_3 = h_4 = \mathbf{0} \tag{20}$$

### 5.2.2 $O$

For this point group symmetry we use the following order parameter and hopping terms.

$$\boldsymbol{l_k} = sin(k_x)\hat{x} + sin(k_y)\hat{y} \tag{21}$$

$$h_0 = \begin{pmatrix} -\mu & 0 & 0 & \Delta_s \\ 0 & -\mu & -\Delta_s & 0 \\ 0 & -\Delta_s^* & \mu & 0 \\ \Delta_s^* & 0 & 0 & \mu \end{pmatrix} \tag{22}$$

$$h_1 = \begin{pmatrix} t/2 & \alpha/(2i) & -\Delta_p/(2i) & 0 \\ \alpha/(2i) & t/2 & 0 & \Delta_p/(2i) \\ -\Delta_p^*/(2i) & 0 & -t/2 & \alpha/(2i) \\ 0 & \Delta_p^*/(2i) & \alpha/(2i) & -t/2 \end{pmatrix} \tag{23}$$

$$h_2 = \begin{pmatrix} t/2 & -\alpha/2 & \Delta_p/2 & 0 \\ \alpha/2 & t/2 & 0 & \Delta_p/2 \\ -\Delta_p^*/2 & 0 & -t/2 & \alpha/2 \\ 0 & -\Delta_p^*/2 & -\alpha/2 & -t/2 \end{pmatrix} \tag{24}$$

$$h_3 = h_4 = \mathbf{0} \tag{25}$$

### 5.2.3 $T_d$

For this point group symmetry we use the following order parameter and hopping terms.

$$\boldsymbol{l_k} = sin(k_x)sin^2(k_y)\hat{x} - sin^2(k_x)sin(k_y)\hat{y} \tag{26}$$

$$h_0 = \begin{pmatrix} -\mu & 0 & 0 & \Delta_s \\ 0 & -\mu & -\Delta_s & 0 \\ 0 & -\Delta_s^* & \mu & 0 \\ \Delta_s^* & 0 & 0 & \mu \end{pmatrix} \tag{27}$$

$$h_1 = \begin{pmatrix} t/2 & \alpha/(4i) & -\Delta_p/(4i) & 0 \\ \alpha/(4i) & t/2 & 0 & \Delta_p/(4i) \\ -\Delta_p^*/(4i) & 0 & -t/2 & \alpha/(4i) \\ 0 & \Delta_p^*/(4i) & \alpha/(4i) & -t/2 \end{pmatrix} \tag{28}$$

$$h_2 = \begin{pmatrix} t/2 & \alpha/4 & -\Delta_p/4 & 0 \\ -\alpha/4 & t/2 & 0 & -\Delta_p/4 \\ \Delta_p^*/4 & 0 & -t/2 & -\alpha/4 \\ 0 & \Delta_p^*/4 & \alpha/4 & -t/2 \end{pmatrix} \tag{29}$$

$$h_3 = \begin{pmatrix} 0 & -\alpha/8 & \Delta_p/8 & 0 \\ \alpha/8 & 0 & 0 & \Delta_p/8 \\ -\Delta_p^*/8 & 0 & 0 & \alpha/8 \\ 0 & -\Delta_p^*/8 & -\alpha/8 & 0 \end{pmatrix} \tag{30}$$

$$h_4 = \begin{pmatrix} 0 & -\alpha/(8i) & \Delta_p/(8i) & 0 \\ -\alpha/(8i) & 0 & 0 & -\Delta_p/(8i) \\ \Delta_p^*/(8i) & 0 & 0 & -\alpha/(8i) \\ 0 & -\Delta_p^*/(8i) & -\alpha/(8i) & 0 \end{pmatrix} \tag{31}$$

### 5.2.4 Implementation Code

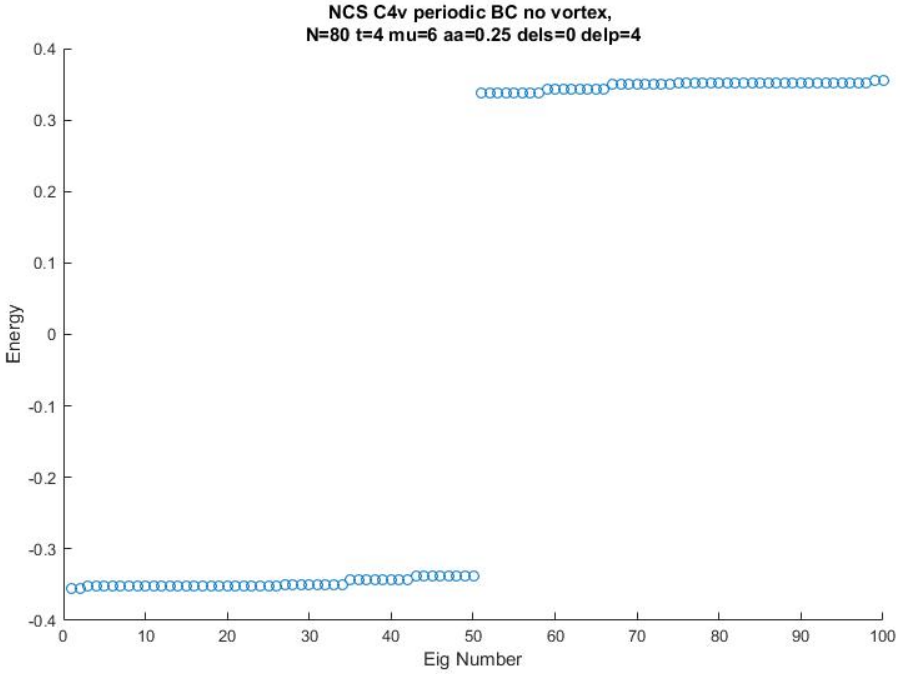
Sample code for the $C_{4v}$ point group is provided in the appendix, the implementation was completely done in MATLAB. The programs used are simple and human readable due to MATLAB's high level abstraction, however in return the calculations are limited to systems with size N = 100 (unless the program is ran on the cluster).

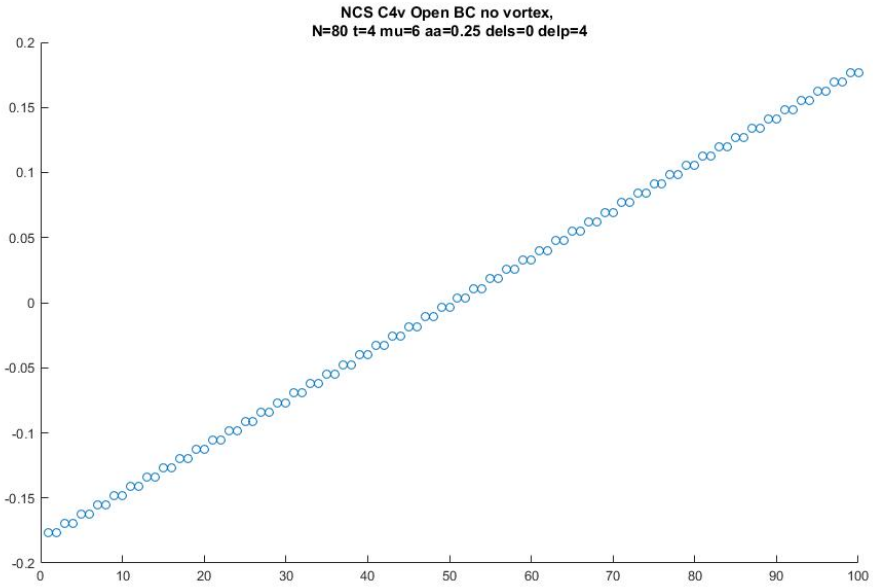
## 5.3 Results

The parameters used in the calculations are $N = 80, t = 4, \mu = 6, \alpha = aa = 0.25, \Delta_s = 0, \Delta_p = 0.4$. We find that $C_{4v}$ and $O$ point group symmetries have zero energy vortex bound states while $T_d$ does not.

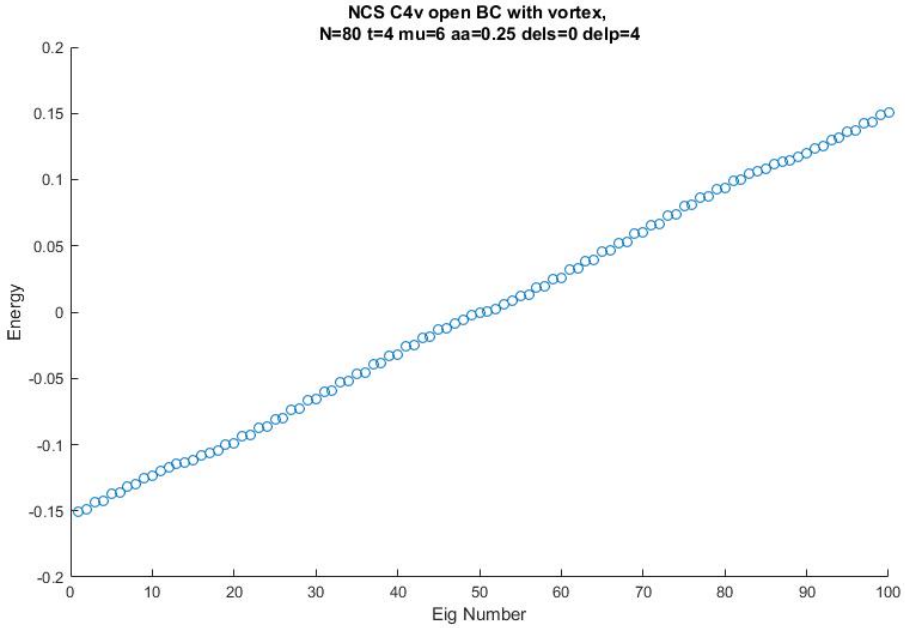
### 5.3.1 $C_{4v}$ With Vortex

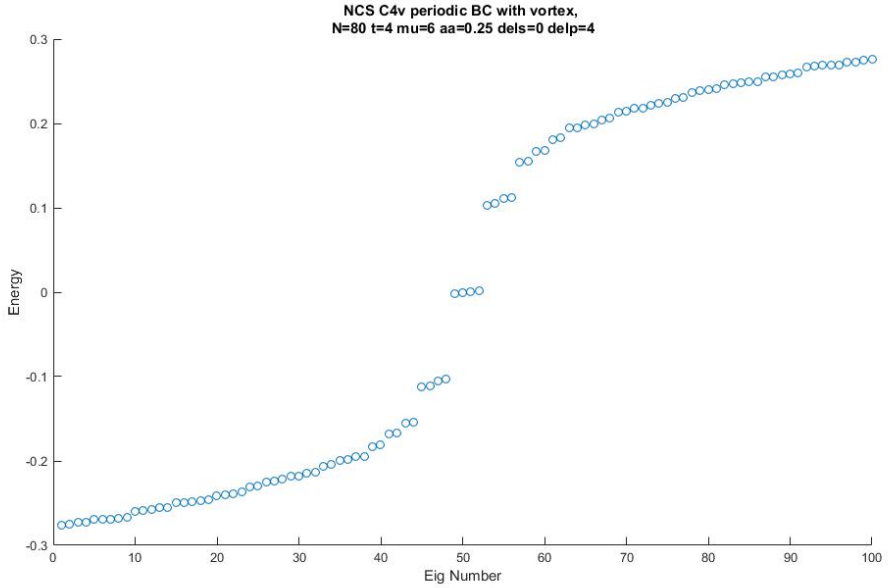
We begin by calculating the spectrum for the model with periodic boundary conditions and no vortex. We observe the expected gap of size $\approx \Delta_p$.
We then run the program with open BC to observe the expected edge states.



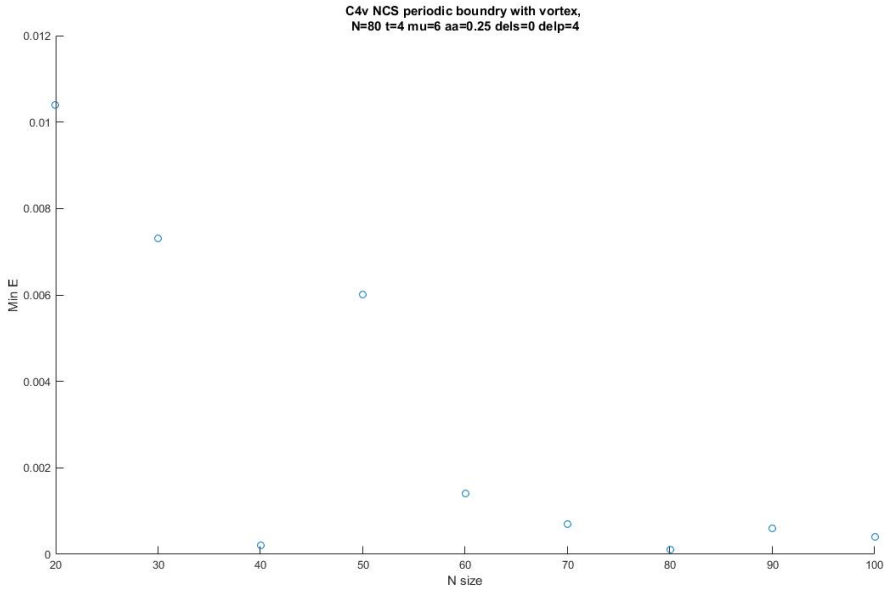




We then implement the vortex with both Open and Periodic BC. We note the zero energy vortex states in the periodic BC system.



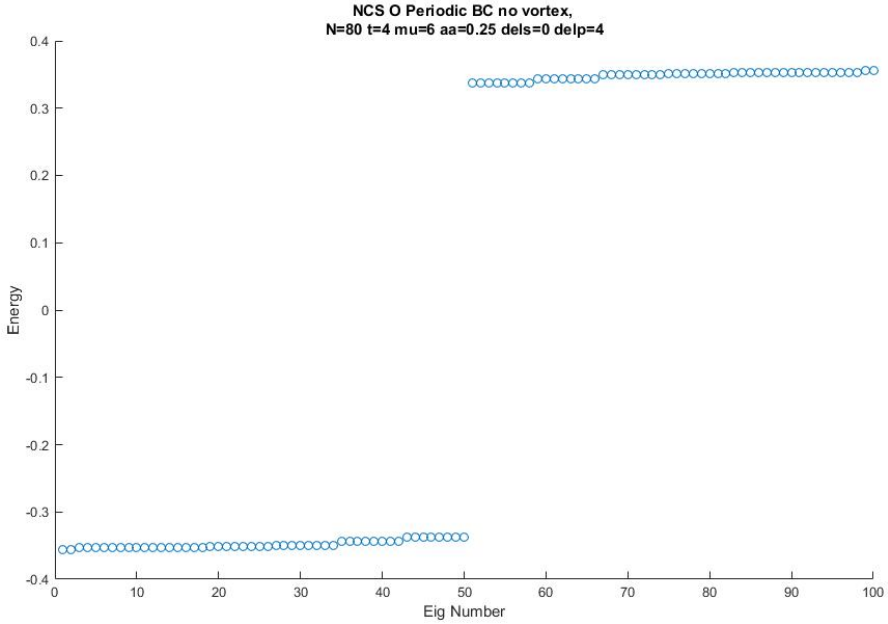

Numerically the zero energy vortex states are on the order of $10^{-4}$, in order to confirm that those states are zero in the thermodynamic limit we calculate the minimum energy in the spectrum for $N = 20 \rightarrow 100$ and plot the system size against the minimum energy value. It is clear that the minimum energy approaches zero as N increases.
We note that at N mod 40 we get dips that do not match the trend, that is a result of

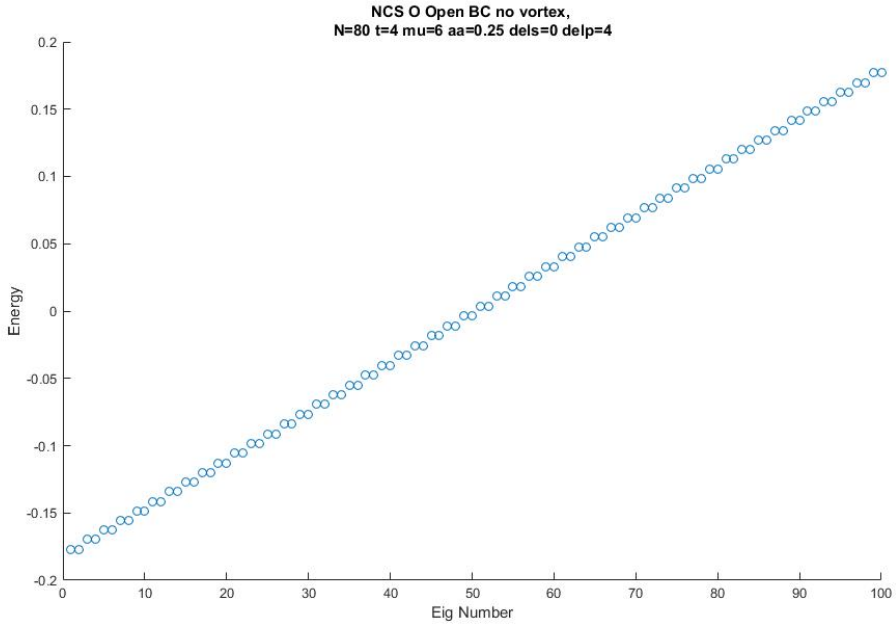


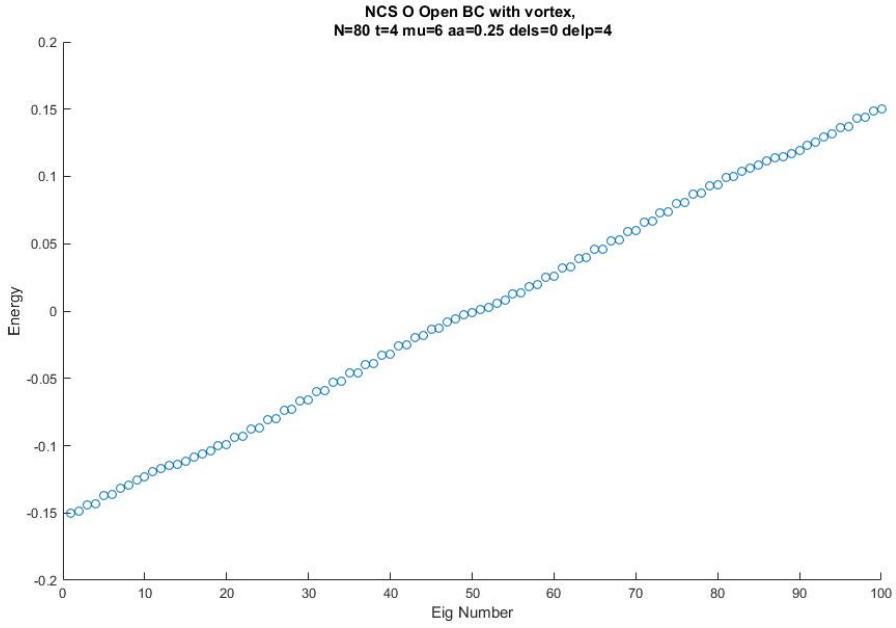

the vortex implementation method, as some random symmetry will arise at mod 40 sized lattices.

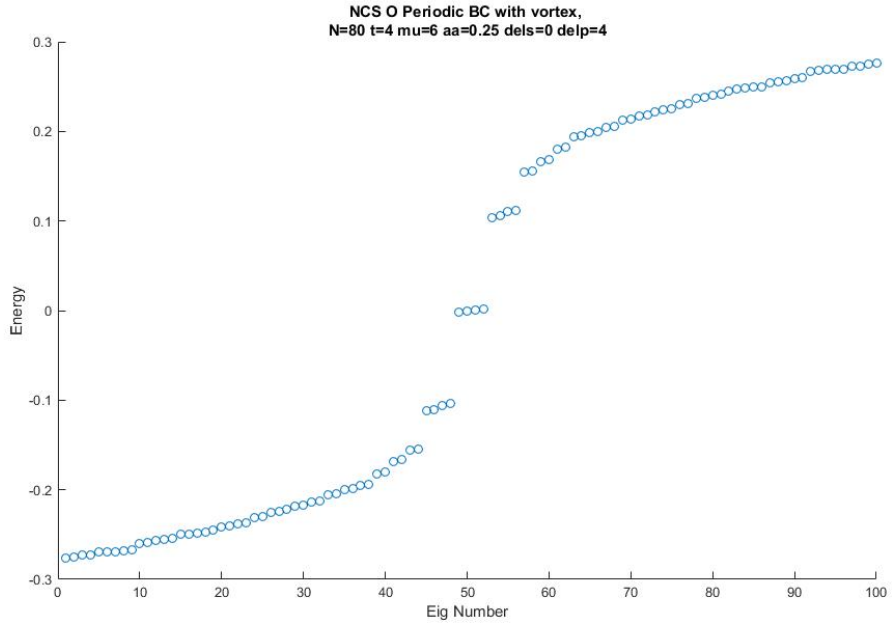
### 5.3.2 *O* With Vortex

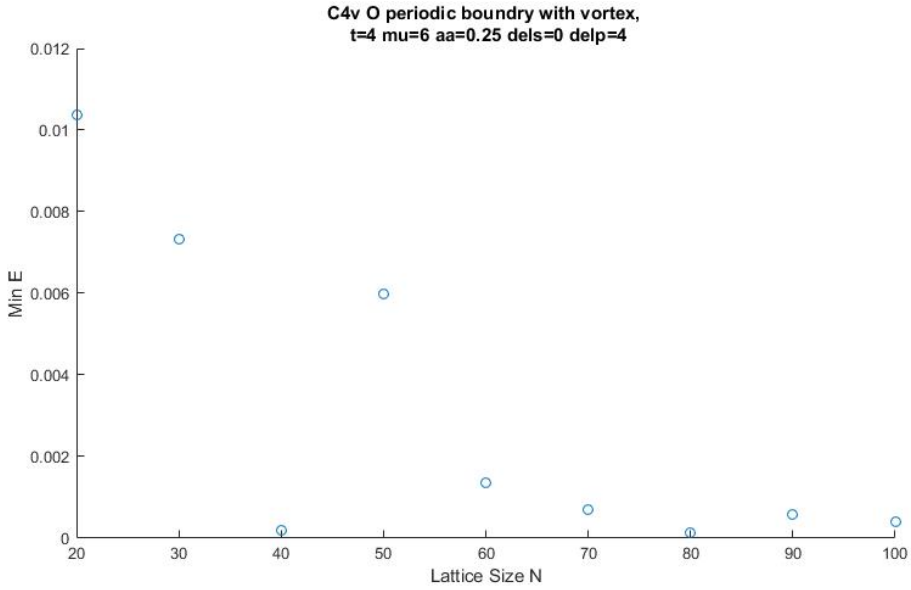
We repeat the same process used on the $C_{4v}$ point group for the $O$ point group. Below are the figures in the same order. We observe identical results to the previous point group.




NCS O Open BC no vortex,
N=80 t=4 mu=6 aa=0.25 dels=0 delp=4
Energy
Eig Number
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
0
10
20
30
40
50
60
70
80
90
100

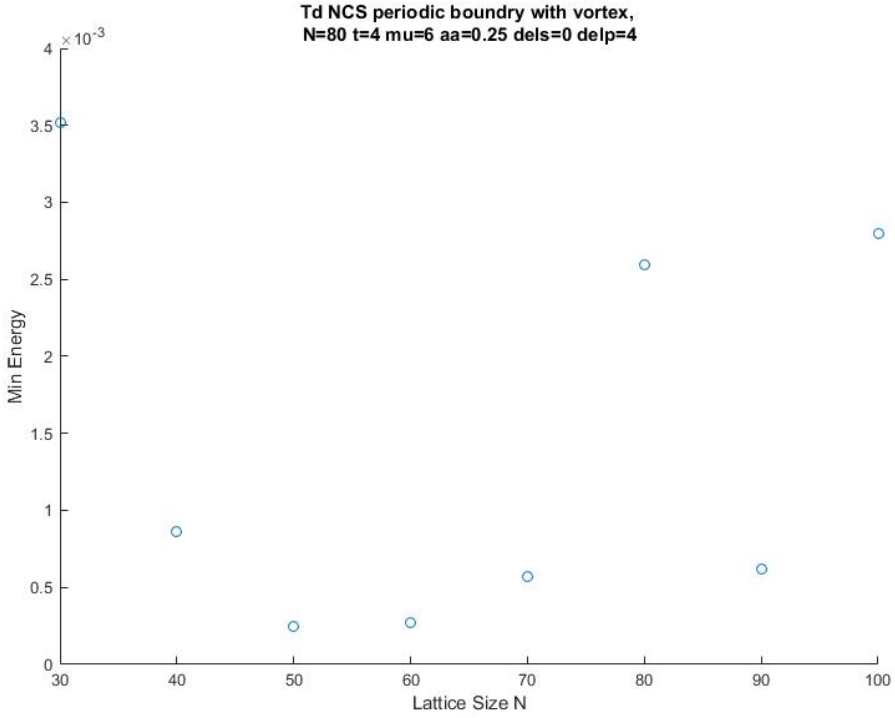

NCS O Open BC with vortex,
N=80 t=4 mu=6 aa=0.25 dels=0 delp=4
Energy
Eig Number
0.2
0.15
0.1
0.05
0
-0.05
-0.1
-0.15
-0.2
0
10
20
30
40
50
60
70
80
90
100


NCS O Periodic BC with vortex,
N=80 t=4 mu=6 aa=0.25 dels=0 delp=4
Energy
Eig Number
0.3
0.2
0.1
0
-0.1
-0.2
-0.3
0
10
20
30
40
50
60
70
80
90
100


C4v O periodic boundry with vortex,
t=4 mu=6 aa=0.25 dels=0 delp=4
Min E
Lattice Size N
0.012
0.01
0.008
0.006
0.004
0.002
0
20
30
40
50
60
70
80
90
100

### 5.3.3 $T_d$ With Vortex

We repeat the same process used on the $T_d$ point group. Below are the figures in the same order. We note that this group does not have zero energy vortex bound states. As N increases minimum energy continues to fluctuate with no trend.



### 5.3.4 $C_{4v}$ With Dislocation

We expect the defect to only induce zero energy states in the topological phase, as a result we begin by calculating the phase diagram for the NCS $C_{4v}$ analytically using the expression

$$E_3 = -E_4 = \sqrt{(\epsilon_{\boldsymbol{k}} - \alpha|\boldsymbol{l_k}|)^2 + (\Delta_s - \Delta_p|\boldsymbol{l_k}|)^2} \tag{32}$$

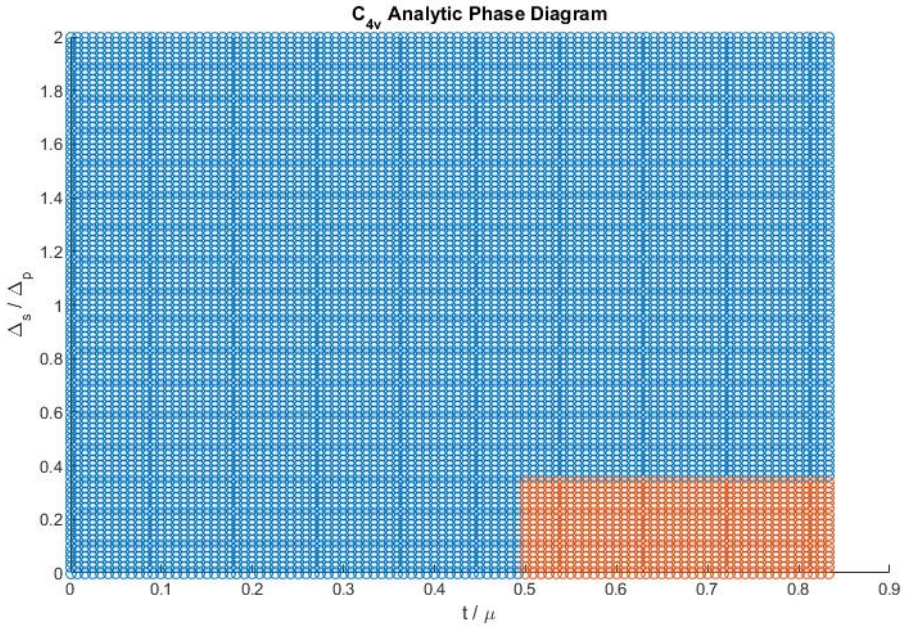



Figure 3: Topological Phase represented by the surface Orange

We then compute the energy spectrum with parameters chosen from the non topological phase, as expected the spectrum is gapped.The spectrum is then recalculated using topological phase parameters, as expected we get dislocation bound zero energy states. Both spectrum can be seen below.

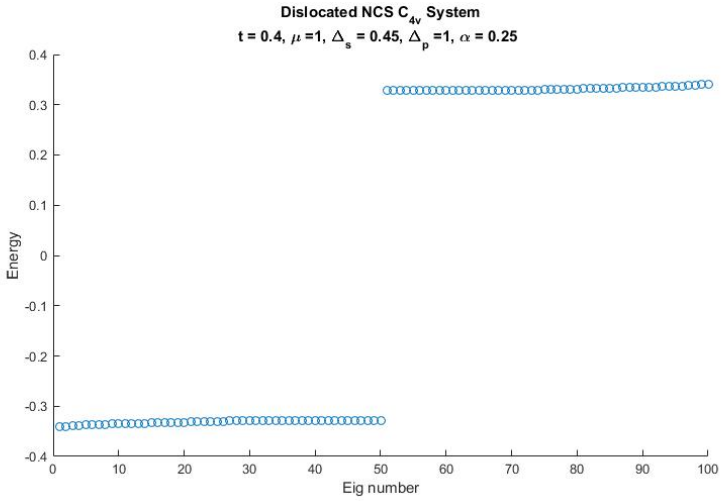



Figure 4: Energy Spectrum in non topological phase

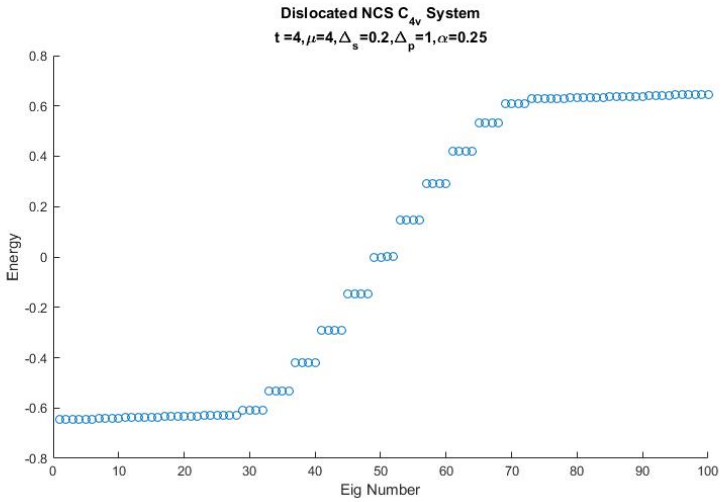



Figure 5: Energy Spectrum in topological phase

Finally we numerically calculate the phase diagram oh the dislocated system, as we can see from the figure below the phase diagram of the dislocated system is in agreement with the analytic phase diagram. The slight differences towards $t/\mu > 0.8$ are the result of the small size of the numerically probed systems. We expect full agreement in the thermodynamic limit.

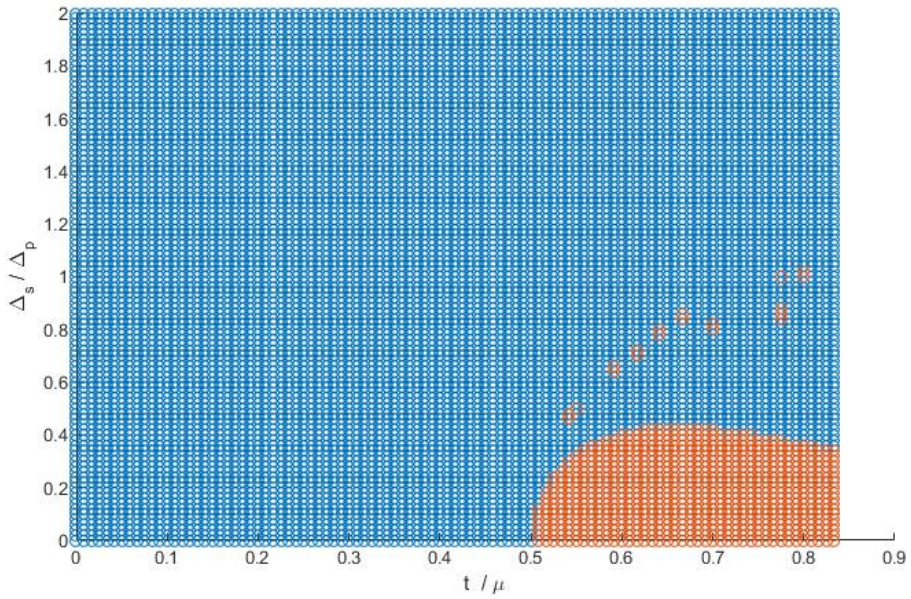



Figure 6: Numerically computed phase diagram with dislocation

For further investigation we calculate the energy spectrum using parameters from the analytically calculated topological phase, the result is the below diagram. The plot is rather unexpected as we only observe zero energy states at $k_z = 0, \pi, -\pi$. I suspect that is a manifestation of the $cos(k_Z)$ term leading t in the on-site terms. It is possible that as $cos(k_z)$ decreases the $t/\mu$ parameter behaves as if pushed beyond the phase boundary along the x-axis of the phase diagram. Further investigation of the topological invariant is needed.

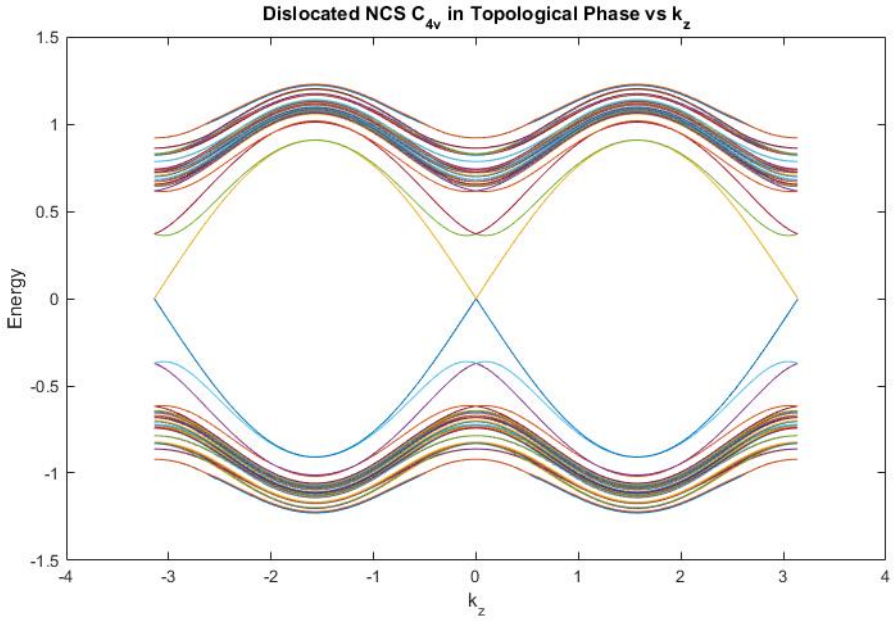



Figure 7: Energy Spectrum against $k_z$

# 6 $f$-Wave Superconductor

## 6.1 Overview

In this section the implementations are all carried out on a 2D triangular lattice. We define the lattice with the following bond vectors:

$$\boldsymbol{T_1} = (0,1) \quad \boldsymbol{T_2} = (\sqrt{3}/2, -1/2) \quad \boldsymbol{T_3} = (-\sqrt{3}/2, -1/2) \tag{33}$$

we also define second neighbour bond vectors as:

$$\boldsymbol{S_1} = (\sqrt{3}/2, 3/2) \quad \boldsymbol{S_2} = (\sqrt{3}, 0) \quad \boldsymbol{S_3} = (\sqrt{3}/2, -3/2) \tag{34}$$

for implementation convenience we convert our vectors from $(x, y) = (i, j)$ real-space to $(x, y) = (T_2, T_1)$ space where the bond vectors are defined as:

$$\boldsymbol{T_1} = (0,1) \quad \boldsymbol{T_2} = (1,0) \quad \boldsymbol{T_3} = (-1,-1) \tag{35}$$

and

$$\boldsymbol{S_1} = (1,2) \quad \boldsymbol{S_2} = (2,1) \quad \boldsymbol{S_3} = (1,-1) \tag{36}$$

where $\tilde{x}$ and $\tilde{y}$ run from -N/2 to N/2. The change of basis matrix is provided below for reference:

$$\begin{pmatrix} 2/\sqrt{3} & 0 \\ 1/\sqrt{3} & 1 \end{pmatrix} \tag{37}$$

Through out this section we will be making 3 different parameter choice combinations based on the recommendation of Dr. Schnyder we will also hold $\Delta_0 = 1$ for convenience, the combinations are as follows:

$$\text{FS1:} \quad t_1 = 1.0, \quad t_2 = 0.0, \quad \mu = -1.3 \tag{38}$$

$$\text{FS2:} \quad t_1 = 1.0, \quad t_2 = 1.0, \quad \mu = -1.5 \tag{39}$$

$$\text{FS3:} \quad t_1 = 1.0, \quad t_2 = 0.0, \quad \mu = 0.0 \tag{40}$$

## 6.2 Definition of Hamiltonian

The $\boldsymbol{k}$-space Hamiltonian if defined as follows $\mathcal{H}(\boldsymbol{k}) = \sum_{\boldsymbol{k}} \psi^\dagger_{\boldsymbol{k}} H(\boldsymbol{k}) \psi_{\boldsymbol{k}}$ with $\psi_{\boldsymbol{k}} = (c_{\boldsymbol{k}\uparrow}, c^\dagger_{-\boldsymbol{k}\downarrow})^T$ where $c_{\boldsymbol{k}\sigma}$ is the electron annihilation operator with spin $\sigma$ and momentum $\boldsymbol{k}$.

$$H(\boldsymbol{k}) = \begin{pmatrix} h(\boldsymbol{k}) & \Delta(\boldsymbol{k}) \\ \Delta^*(\boldsymbol{k}) & -h(\boldsymbol{k}) \end{pmatrix} \tag{41}$$

where

$$h(\boldsymbol{k}) = \epsilon_{1\boldsymbol{k}} + \epsilon_{2\boldsymbol{k}} - \mu \tag{42}$$

$$\Delta(\boldsymbol{k}) = \Delta_0 \Big( sin(k_y) - 2cos(\sqrt{3}k_x/2) sin(k_y/2) \Big) \tag{43}$$

and

$$\epsilon_{1\boldsymbol{k}} = t_1 (2cos(\sqrt{3}k_x/2) cos(k_y/2) + cos(k_y)) \tag{44}$$

$$\epsilon_{2\boldsymbol{k}} = t_2 (2cos(\sqrt{3}k_x/2) cos(3k_y/2) + cos(\sqrt{3}k_x)) \tag{45}$$

## 6.3 Implementation of Real-Space Hamiltonian

To implement the Hamiltonian it is first converted to real space by taking the Discrete Fourier Transform. The full Hamiltonian $H$ is identical to the one implemented for NCS (equation(7)).

$$H_0 = \begin{pmatrix} T_0 & T_2^\dagger & & & T_2 \\ T_2 & T_0 & T_2^\dagger & & \\ & T_2 & T_0 & \ddots & \\ & & \ddots & \ddots & \\ T_2^\dagger & & & & \end{pmatrix} \tag{46}$$

$$H_1 = \begin{pmatrix} T_1^\dagger & T_3 & S_2^\dagger & & 0 & S_3 \\ S_3 & T_1^\dagger & T_3 & \ddots & & 0 \\ 0 & S_3 & T_1^\dagger & \ddots & & \\ & \ddots & \ddots & T_1^\dagger & & \\ S_2^\dagger & & & & \ddots & \\ T_3 & S_2^\dagger & & & & \end{pmatrix} \tag{47}$$

$$H_2 = \begin{pmatrix} 0 & S_1^\dagger & & & 0 \\ & 0 & S_1^\dagger & & \\ & & 0 & & \\ S_1^\dagger & & \ddots & 0 & \\ 0 & S_1^\dagger & & & \ddots \end{pmatrix} \tag{48}$$

$$T_0 = \begin{pmatrix} -\mu & 0 \\ 0 & \mu \end{pmatrix} \quad T_1 = \begin{pmatrix} t_1 & i\Delta_0 \\ i\Delta_0^* & -t_1 \end{pmatrix} \tag{49}$$

$$T_2 = \begin{pmatrix} 4t_1 & 4i\Delta_0 \\ 4i\Delta_0^* & -4t_1 \end{pmatrix} \quad T_3 = \begin{pmatrix} 4t_1 & 4i\Delta_0 \\ 4i\Delta_0^* & -4t_1 \end{pmatrix} \tag{50}$$

$$S_1 = \begin{pmatrix} 4t_2 & 0 \\ 0 & -4t_2 \end{pmatrix} \quad S_2 = \begin{pmatrix} t_2 & 0 \\ 0 & -t_2 \end{pmatrix} \quad S_3 = \begin{pmatrix} 4t_2 & 0 \\ 0 & -4t_2 \end{pmatrix} \tag{51}$$

## 6.4 Implementation of Mixed-Space Hamiltonian

A defect free triangular lattice has translational symmetry in the defined x and y direction, as a result $k_x$ and $k_y$ are well behaved quantum numbers in the brillouin zone allowing us to use the mixed space representations (x,$k_y$) and ($k_x$,y) to compute the single dimension

edge states. The problem is then reduced to diagonalizing a 2N by 2N matrix for every $k_i$ in the brioullin zone. The general 2N-by-2N is given by:

$$H_0 = \begin{pmatrix} T_0 & T_1 & T_2 & T_3 & 0 \\ T_1 & T_0 & T_1 & T_2 & \\ T_2 & T_1 & T_0 & \ddots & \\ T_3 & T_2 & \ddots & \ddots & \\ 0 & & & & \end{pmatrix} \tag{52}$$

### 6.4.1 Periodic y case: (x,$k_y$)

$$T_0 = \sqrt{2\pi} \begin{pmatrix} -\mu + t_1 cos(k_y) & \Delta_0 sin(k_y) \\ \Delta_0^* sin(k_y) & \mu - t_1 cos(k_y) \end{pmatrix} \tag{53}$$

$$T_1 = \begin{pmatrix} \eta & \zeta \\ \zeta^* & -\eta \end{pmatrix} \tag{54}$$

$$T_2 = \begin{pmatrix} t_2\sqrt{\pi/2} & 0 \\ 0 & -t_2\sqrt{\pi/2} \end{pmatrix} \tag{55}$$

$$T_3 = 0 \tag{56}$$

where $\eta$ and $\zeta$ are defined for convenience as

$$\eta = t_1(2\sqrt{2\pi}\cos(k_y/2)) + t_2(\sqrt{\pi/2} \cdot 4\cos(3k_y/2)) \tag{57}$$

$$\zeta = \Delta_0(-2\sqrt{2\pi} \cdot \sin(k_y/2)) \tag{58}$$

### 6.4.2 Periodic x case: ($k_x$,y)

$$T_0 = 2\sqrt{\pi/2} \begin{pmatrix} -\mu + t_2 cos(\sqrt{3}k_x) & 0 \\ 0 & \mu - t_2 cos(\sqrt{3}k_x) \end{pmatrix} \tag{59}$$

$$T_1 = \begin{pmatrix} t_1(\sqrt{\pi/2} \cdot 4cos(\sqrt{3}k_x/2)) & \Delta_0(i\sqrt{\pi/2} \cdot 4cos(\sqrt{3}k_x/2)) \\ \Delta_0^*(i\sqrt{\pi/2} \cdot 4cos(\sqrt{3}k_x/2)) & -t_1(\sqrt{\pi/2} \cdot 4cos(\sqrt{3}k_x/2)) \end{pmatrix} \tag{60}$$

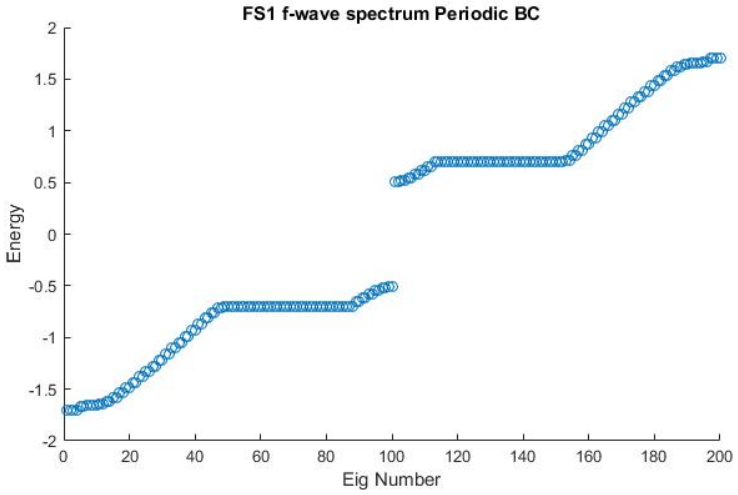
$$T_2 = \begin{pmatrix} t_1\sqrt{\pi/2} & -i\Delta_0\sqrt{\pi/2} \\ -i\Delta_0^*\sqrt{\pi/2} & -t_1\sqrt{\pi/2} \end{pmatrix} \tag{61}$$

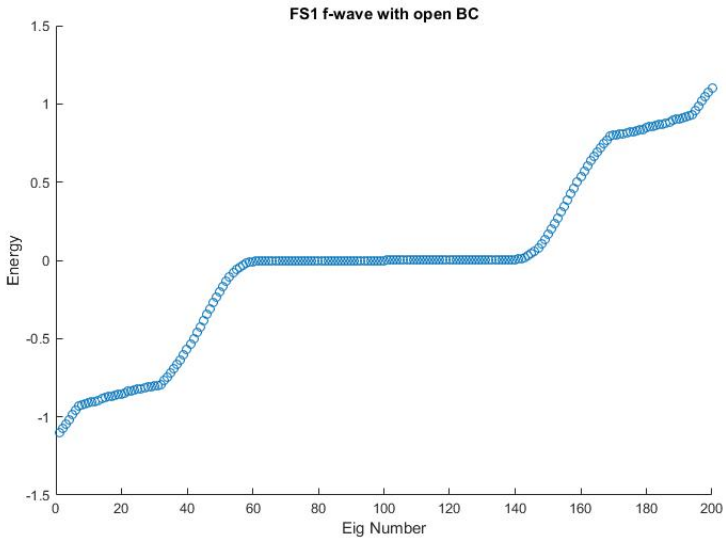
$$T_3 = \begin{pmatrix} t_2\sqrt{\pi/2} & 0 \\ 0 & -t_2\sqrt{\pi/2} \end{pmatrix} \tag{62}$$

## 6.5 Results

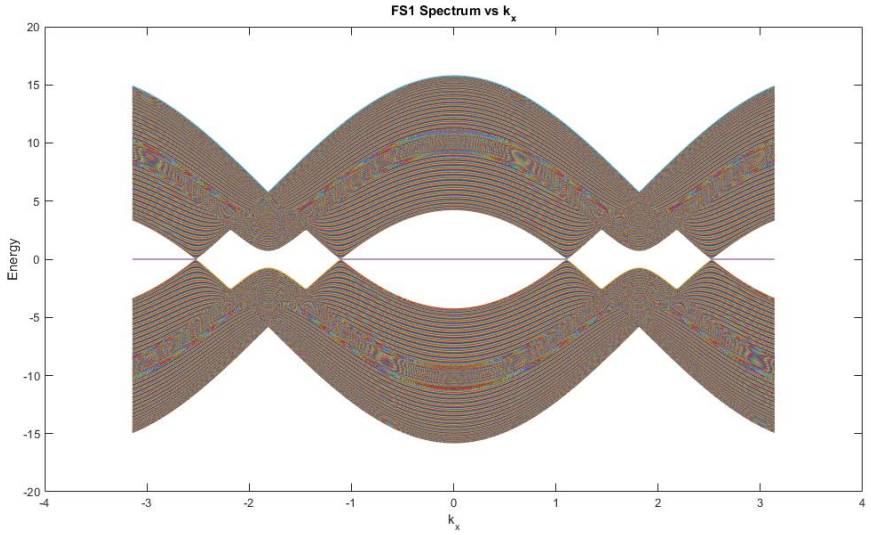
### 6.5.1 Edge-States

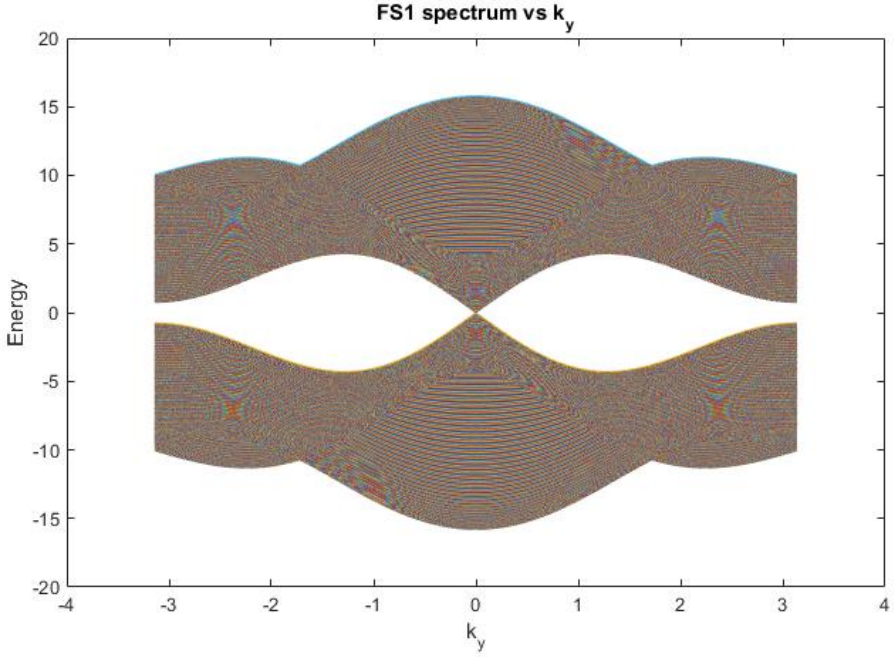
We begin by investigating the FS1 parameters. First we calculate the spectrum in periodic conditions. We observe the predicted gap.We then recalculate with open boundary conditions and we observe the appearance of zero energy edge states.



We then used mixed representation to calculate the spectrums as a function of well defined $k_x$ and $k_y$. The results are below and agree with predictions and with the real space results. The presence of zero energy states is very clear from the figures. (Single edge real-space figures avilable in the appendix)



The Process is the repeated with FS2 and FS3. The resulting figures are presented in the appendix.

### 6.5.2 Strain

To be completed (See section 7 further work)

# 7 Further Work

Although the nature of my upcoming work my face slight changes, the goal will remain to investigate defect induced flat-bands and their properties. Some specific tasks include:

1. Implement triaxial strain defect on the f-wave triangular lattice expecting to observe the induction of a synthetic magnetic field which will result in Landau levels of Majorana Fermions.
2. Investigate the absence of zero energy states in a dislocated NCS $C_{4v}$ for $k_Z$ between 0 and $\pi$.
3. Expand the investigation to include investigating defects in semimetals and examine the resultant bound states.

# 8 Appendix

## 8.1 NCS Example Program

```
function [ V,EN ] = NCS_C4v( N, t, mu, alpha, deltas, deltap, kz, vortex, periodicx, periodicy,
    nEV )

%==========================================================================
% Constants

a1 = 1;
a2 = 1;
d = 4; % dimention of spin orbit terms

% Vortex parameters
c = 1;
w = 1;
a = (N/4); % Seperation that maximizes distance between vortices

% Ensure coninuity by finding optimal alpha and beta based on the minmization
% Credit: Derrick previous work
diff_fun = @(x) dirdiff(x(1),x2,N,w);
[x,fval] = fminsearch(diff_fun,[0.01,2]);
AA = x(1);
BB = x2;

% Construct the full Hamiltonian, H
H = zeros(d*N^2);
H = sparse(H); % sparse => 0-> no memory

% constant matrix outside of the loop for optimization
h0a = diag([-mu, -mu, mu, mu]);

%==========================================================================

% for each H0,H1 the y is held constant and x varys
for ny = 1:N

    H0 = zeros(d*N);
    H1 = zeros(d*N);

    y = -N/2 + (ny-1) - 0.5;

    for nx = 1:N
        % Implement on the onsite component
        x = -N/2 + (nx-1) - 0.5;
        phi = atan2((x^2 + y^2 - a^2), 2*a*y*(abs(1+AA-abs(y)/(N/2)))^(1/BB) );
        r1 = sqrt((x-a)^2+y^2);
```

```
        r2 = sqrt((x+a)^2+y^2);

        if(vortex == 1)
            deltaS = deltas * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
            deltaP = deltap * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
        else
            deltaS = deltas;
            deltaP = deltap;
        end

        h0b = fliplr(diag([deltaS + deltaP*sin(kz), - deltaS + deltaP*sin(kz), ...
            -conj(deltaS) + conj(deltaP)*sin(kz), conj(deltaS) + conj(deltaP)*sin(kz)]));
        h0 = h0a + h0b;

        % H0 diagonal h0 terms y = y and x in [-N/2,N/2)
        H0(d*(nx-1)+1:d*nx, d*(nx-1)+1:d*nx) = h0;


         if(nx~=1)
            % Calculate values for the vortex
            x = -N/2 + (nx-1) - 0.5; % add 0.5 because this is in between sites
            phi = atan2((x^2 + y^2 - a^2), 2*a*y*(abs(1+AA-abs(y)/(N/2)))^(1/BB) );
            r1 = sqrt((x-a)^2+y^2);
            r2 = sqrt((x+a)^2+y^2);

            if(vortex == 1)
                deltaP = deltap * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
            else
                deltaP = deltap;
            end

            h1 = [ t/2, -a1*alpha/(2*1i), a1*deltaP/(2*1i), 0; ...
                -a1*alpha/(2*1i), t/2, 0, -a1*deltaP/(2*1i); ...
                conj(a1*deltaP)/(2*1i), 0, -t/2, -a1*alpha/(2*1i); ...
                0, -conj(a1*deltaP)/(2*1i), -a1*alpha/(2*1i), -t/2];

            H0(d*(nx-1)+1:d*nx, d*(nx-2)+1:d*(nx-1)) = h1';
            H0(d*(nx-2)+1:d*(nx-1), d*(nx-1)+1:d*nx) = h1;
        end

    end
%=========================================================================

% Create the last two blocks for periodic BC in x-direction
    if(periodicx == 1)
        x = -N/2 - 0.5;
        phi = atan2((x^2 + y^2 - a^2), 2*a*y*(abs(1+AA-abs(y)/(N/2)))^(1/BB) );
        r1 = sqrt((x-a)^2+y^2);
        r2 = sqrt((x+a)^2+y^2);

        if(vortex == 1)
            deltaP = deltap * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
        else
            deltaP = deltap;
        end

        h1 = [ t/2, -a1*alpha/(2*1i), a1*deltaP/(2*1i), 0; ...
            -a1*alpha/(2*1i), t/2, 0, -a1*deltaP/(2*1i); ...
            conj(a1*deltaP)/(2*1i), 0, -t/2, -a1*alpha/(2*1i); ...
            0, -conj(a1*deltaP)/(2*1i), -a1*alpha/(2*1i), -t/2];

        H0(1:4,end-3:end) = h1';
        H0(end-3:end,1:4) = h1;
    end

    H(d*N*(ny-1)+1:d*N*(ny),d*N*(ny-1)+1:d*N*(ny)) = H0;
%=========================================================================

% Make off diagonal block
    if(ny ~= 1) % No diagonal block on the first iteration
        y = -N/2 + (ny-1) - 0.5; % range of y is [-N/2, N/2)
        for nx = 1:N
            x = -N/2 + (nx-1);
            phi = atan2((x^2 + y^2 - a^2), 2*a*y*(abs(1+AA-abs(y)/(N/2)))^(1/BB) );
            r1 = sqrt((x-a)^2+y^2);
            r2 = sqrt((x+a)^2+y^2);

            if(vortex == 1)
                deltaP = deltap * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
            else
                deltaP = deltap;
            end

            h2 = [ t/2, a2*alpha/2, -a2*deltaP/2, 0; ...
                -a2*alpha/2, t/2, 0, -a2*deltaP/2; ...
                conj(a2*deltaP)/2, 0, -t/2, -a2*alpha/2; ...
                0, conj(a2*deltaP)/2, a2*alpha/2, -t/2];

            H1(4*(nx-1)+1:4*nx,4*(nx-1)+1:4*nx) = h2;
        end

        H(d*N*(ny-1)+1:d*N*(ny),d*N*(ny-2)+1:d*N*(ny-1)) = H1';
        H(d*N*(ny-2)+1:d*N*(ny-1),d*N*(ny-1)+1:d*N*(ny)) = H1;
    end
```

```
end
%=========================================================================

% Create the last two blocks for periodic BC in y-direction
if(periodicy == 1)
    y = -N/2 - 0.5;
    for nx = 1:N
        x = -N/2 + (nx-1) - 0.5;
        phi = atan2((x^2 + y^2 - a^2), 2*a*y*(abs(1+AA-abs(y)/(N/2)))^(1/BB) );
        r1 = sqrt((x-a)^2+y^2);
        r2 = sqrt((x+a)^2+y^2);

        if(vortex == 1)
            deltaP = deltap * tanh(r1/c) * tanh(r2/c) * exp(1i*w*phi);
        else
            deltaP = deltap;
        end

        h2 = [ t/2, a2*alpha/2, -a2*deltaP/2, 0; ...
            -a2*alpha/2, t/2, 0, -a2*deltaP/2; ...
            conj(a2*deltaP)/2, 0, -t/2, -a2*alpha/2; ...
            0, conj(a2*deltaP)/2, a2*alpha/2, -t/2];

        H1(4*(nx-1)+1:4*nx,4*(nx-1)+1:4*nx) = h2;
    end

    H(1:4*N, end-(4*N-1):end) = H1';
    H(end-(4*N-1):end, 1:4*N) = H1;
end
%=========================================================================

% Return the nEv eigvals and eigvects closest to 0
[V,E] = eigs(H,nEV,'sm');
EN = diag(E);

end
```

## 8.2 Dislocation Example Program

```
function [ V,EN ] = Dislocate_out_NCS_C4v( N, t, mu, alpha, deltaS, deltaP,kz, periodicx,
    periodicy, nEV )
%Dislocations are all introduces along the x axis Ensure that N is odd
a1=1;
a2=1;
d=4;
%Construct onsite and first hopping matrices (dxd)
h0a = diag([t*cos(kz)+alpha*kz-mu, t*cos(kz)-alpha*kz-mu, -t*cos(kz)+alpha*kz+mu, -t*cos(kz)-
    alpha*kz+mu]);
h0b = fliplr(diag([deltaS + deltaP*sin(kz), - deltaS + deltaP*sin(kz), ...
          -conj(deltaS) + conj(deltaP)*sin(kz), conj(deltaS) + conj(deltaP)*sin(kz)]));
h0 = h0a + h0b;

h1 = [ t/2, -a1*alpha/(2*1i), a1*deltaP/(2*1i), 0; ...
   -a1*alpha/(2*1i), t/2, 0, -a1*deltaP/(2*1i); ...
    conj(a1*deltaP)/(2*1i), 0, -t/2, -a1*alpha/(2*1i); ...
    0, -conj(a1*deltaP)/(2*1i), -a1*alpha/(2*1i), -t/2];

h2 = [ t/2, a2*alpha/2, -a2*deltaP/2, 0; ...
   -a2*alpha/2, t/2, 0, -a2*deltaP/2; ...
    conj(a2*deltaP)/2, 0, -t/2, -a2*alpha/2; ...
    0, conj(a2*deltaP)/2, a2*alpha/2, -t/2];

%=========================================================
% Create dislocation,bulk and full hamitlonains
Hdis = M_NCS_kz( N-1, t, mu, alpha, deltaS, deltaP, kz, 1, 0, 0, periodicy, 0 );
Hbulk = M_NCS_kz( N, t, mu, alpha, deltaS, deltaP, kz, 1, 0, 0, periodicy, 0 );

H = blkdiag(Hdis,Hbulk,Hbulk,Hdis);
H = sparse(H);
%=========================================================

for ny = 1:(N-1)
    for nx = 1:(N-1)

        %handle left dislocation
        if(1<=nx<=N-1)

            %if nx = N-1 and ny > half ... hop 2 bulk

            if(nx == (N-1))
                if(ny <= (N-1)/2)
                    %^^ add h1 to row = (N-1,y) col =(N,y) and hc
                    row = ((ny-1)*(N-1)+N-1);
                    col = ((N-1)^2 + (ny-1)*N + 1);

                    H((row-1)*d+1:row*d,(col-1)*d+1:col*d) = h1;
                    H((col-1)*d+1:col*d,(row-1)*d+1:row*d) = h1';

                    %^^ add h1+ to row = (3N,y) col =(3N-1,y) and hc
                    row = (N-1)^2+2*N^2+(ny-1)*(N-1)+1;
                    col = (N-1)^2+N^2+(ny-1)*N+N;

                    H((row-1)*d+1:row*d,(col-1)*d+1:col*d) = h1';
                    H((col-1)*d+1:col*d,(row-1)*d+1:row*d) = h1;
                end
                if(ny > (N-1)/2)
                    %^^ add h1 to row = (N-1,y) col =(N,y+1) and hc
                    row = ((ny-1)*(N-1)+N-1);
                    col = ((N-1)^2 + (ny)*N + 1);

                    H((row-1)*d+1:row*d,(col-1)*d+1:col*d) = h1;
                    H((col-1)*d+1:col*d,(row-1)*d+1:row*d) = h1';
                    %^^ add h1+ to row = (3N,y) col =(3N-1,y+1) and hc
                    row = (N-1)^2+2*N^2+(ny-1)*(N-1)+1;
                    col = (N-1)^2+N^2+(ny)*N+N;

                    H((row-1)*d+1:row*d,(col-1)*d+1:col*d) = h1';
                    H((col-1)*d+1:col*d,(row-1)*d+1:row*d) = h1;
                end
            end

            if(periodicx == 1)
                % add h1 to row=(4N-2,y) col=(1,y) and hc
                row = (N-1)^2+2*N^2+(ny-1)*(N-1)+(N-1);
                col = (ny-1)*(N-1)+1;

                H((row-1)*d+1:row*d,(col-1)*d+1:col*d) = h1;
                H((col-1)*d+1:col*d,(row-1)*d+1:row*d) = h1';
            end

        end

    end
end

%=========================================================
% Handle coupling between dis and bulk

%=========================================================
% Handle periodic x
if(periodicx == 1)
```

```
end

% Return the nEv eigvals and eigvects closest to 0
[V,E] = eigs(H,nEV,'sm');
EN = diag(E);
end
```

## 8.3 $f$-wave Example Program

```
function [ V,EN ] = f_wave(N, t1, t2, mu, delt0, periodicx,periodicy, nEV)

% Dimension of the matrix (2 or 4 depending on the presence of spin orbit coupling)
d = 2;

% Construct full real space hamiltonian
H = zeros(d*N^2);
%H = sparse(H);
for ny = 1:N

    H0 = zeros(d*N);
    H1 = zeros(d*N);
    H2 = zeros(d*N);

    T0 = diag([-mu,mu]);

    %y = !!!;

    %=========================================================================
    % Handle H0 (on-site and hopping T2 ) and periodic x
    for nx = 1:N

        H0(d*(nx-1)+1:d*nx, d*(nx-1)+1:d*nx) = T0;

        if(nx~=1)

            T2 = [4*t1, -4*1i*delt0;...
                -4*1i*conj(delt0), -4*t1];

            H0(d*(nx-1)+1:d*nx, d*(nx-2)+1:d*(nx-1)) = T2;
            H0(d*(nx-2)+1:d*(nx-1), d*(nx-1)+1:d*nx) = T2';
        end
    end

    if(periodicx == 1)
        H0(1:d,end-(d-1):end) = T2;
        H0(end-(d-1):end,1:d) = T2';
    end

    H(d*N*(ny-1)+1:d*N*(ny),d*N*(ny-1)+1:d*N*(ny)) = H0;

    %=========================================================================
    % Handle H1 (hopping T1) and periodic x
    if(ny ~= 1) % No diagonal block on the first iteration

        for nx = 1:N

            T1 = [t1, -1i*delt0; ...
                -1i*conj(delt0), -t1];

            H1(d*(nx-1)+1:d*nx, d*(nx-1)+1:d*nx) = T1';

            if(nx~=1)

                T3 = [4*t1, -4*1i*delt0;...
                        -4*1i*conj(delt0),-4*t1];
                S3 = [4*t2, 0;...
                        0, -4*t2];
                S2 = [t2, 0;...
                        0, -t2];

                H1(d*(nx-1)+1:d*nx, d*(nx-2)+1:d*(nx-1)) = T3;
                H1(d*(nx-2)+1:d*(nx-1), d*(nx-1)+1:d*nx) = S3;
                if (nx~=N)
                    H1(d*(nx)+1:d*(nx+1),d*(nx-2)+1:d*(nx-1)) = S2';
                end
            end
        end

        if(periodicx == 1)
            H1(1:d,end-(d-1):end) = T3;
            H1(end-(d-1):end,1:d) = S3;

            H1(1:d,end-(2*d-1):end-d) = S2';
            H1(d+1:2*d,end-(d-1):end) = S2';
        end

        H(d*N*(ny-1)+1:d*N*(ny),d*N*(ny-2)+1:d*N*(ny-1)) = H1';
        H(d*N*(ny-2)+1:d*N*(ny-1),d*N*(ny-1)+1:d*N*(ny)) = H1;

    end

    %=========================================================================
```

```
    % Handle H2 (hopping 2*T1 ) and periodic x
    if(ny~=1 && ny~=N)
        for nx = 1:N

            if(nx~=1)
                S1 = [4*t2, 0;...
                    0,-4*t2];
                H2(d*(nx-1)+1:d*nx,d*(nx-2)+1:d*(nx-1)) = S1';
            end
        end

        if(periodicx == 1)
            H2(1:d,end-(2*d-1):end-d) = S1';
            H2(d+1:2*d,end-(d-1):end) = S1';
        end

        H(d*N*(ny-2)+1:d*N*(ny-1),d*N*(ny)+1:d*N*(ny+1)) = H2;
        H(d*N*(ny)+1:d*N*(ny+1),d*N*(ny-2)+1:d*N*(ny-1)) = H2';
    end

end

%=========================================Periodic y=========================================
if(periodicy ==1)
    %=============================================H1=============================================
    for nx = 1:N

        T1 = [t1, -1i*delt0; ...
            -1i*conj(delt0), -t1];

        H1(d*(nx-1)+1:d*nx, d*(nx-1)+1:d*nx) = T1';

        if(nx~=1)

            T3 = [4*t1, -4*1i*delt0;...
                -4*1i*conj(delt0),-4*t1];
            S3 = [4*t2, 0;...
                0, -4*t2];
            S2 = [t2, 0;...
                0, -t2];

            H1(d*(nx-1)+1:d*nx, d*(nx-2)+1:d*(nx-1)) = T3;
            H1(d*(nx-2)+1:d*(nx-1), d*(nx-1)+1:d*nx) = S3;
            if (nx~=N)
                H1(d*(nx)+1:d*(nx+1),d*(nx-2)+1:d*(nx-1)) = S2';
            end
        end
    end

    if(periodicx == 1)
        H1(1:d,end-(d-1):end) = T3;
        H1(end-(d-1):end,1:d) = S3;

        H1(1:d,end-(2*d-1):end-d) = S2';
        H1(d+1:2*d,end-(d-1):end) = S2';
    end

    H(1:N*d,end-(N*d-1):end) = H1';
    H(end-(N*d-1):end,1:N*d) = H1;
    %=============================================H2=============================================
    for nx = 1:N

        if(nx~=1)
            S1 = [4*t2, 0;...
                0,-4*t2];
            H2(d*(nx-1)+1:d*nx,d*(nx-2)+1:d*(nx-1)) = S1';
        end
    end

    if(periodicx == 1)
        H2(1:d,end-(2*d-1):end-d) = S1';
        H2(d+1:2*d,end-(d-1):end) = S1';
    end

    H(1:d*N,end-(2*d*N-1):end-d*N) = H2';
    H(d*N+1:2*d*N,end-(d*N-1):end) = H2';

    H(end-(2*d*N-1):end-d*N,1:d*N) = H2;
    H(end-(d*N-1):end,d*N+1:2*d*N) = H2;
end

% Calculate the eigen values and vectors of the hamiltonian
%[V,E] = eigs(H,nEV,'sm');
[V,E] = eig(H);
EN = diag(E);

end
```

## 8.4 Additional Figures

# References


[1] C.-K. Chiu, J. C. Y. Teo, A. P. Schnyder, and S. Ryu. Classification of topological quantum matter with symmetries. *ArXiv e-prints*, May 2015.

[2] M. Z. Hasan and C. L. Kane. *Colloquium* : Topological insulators. *Rev. Mod. Phys.*, 82:3045–3067, Nov 2010.

[3] D. Lee. August Coop Report Analysis of Bound States in Defects in Superconductors.

[4] K. Samokhin. Spinorbit coupling and semiclassical electron dynamics in noncentrosymmetric metals. *Annals of Physics*, 324(11):2385 – 2407, 2009.

[5] M. Sato and Y. Ando. Topological Superconductors. *ArXiv e-prints*, Aug. 2016.

[6] A. P. Schnyder and S. Ryu. Topological phases and surface flat bands in superconductors without inversion symmetry. , 84(6):060504, Aug. 2011.