学习记录

Bo Zhao (赵博) (Dated: June 21, 2024)

蓝色是以后会用的/需要注意的,红色是待完成项。

手上压的学习任务有点多,之后先专心过基础知识,之后有多余时间把之前看的几篇文章好好整理一下,之后顺着看一下Wave-Function Tomography of Topological Dimer Chains with Long-Range Couplings,这篇应该和之前看的声子晶体Large winding number比较相似,顺便可以尝试一下用comsol复现。都了结之后做comsol官网声子晶体的案例(包含色散关系),尝试复现Large winding number仿真部分。看一下Ref. [1] (Realization of fractional quantum Hall state with interacting photons)。

2024/06/20 2024/06/21

复现宇宸师兄在群里发的文章(基于紧束缚模型求解螺旋波导阵列)Ref. [2]. Ref. [2]是Nature文章Photonic Floquet topological insulators拓扑理论部分的学习笔记 [3]。Ref. [3]的亮点在于: 1. 光波段实现(基于打破时间反演)。2. 新的拓扑绝缘体实现方式。3. 无磁,更易小型化集成化(作者没说,我认为的)。螺旋波导电磁理论: Ref. [4]。拓扑绝缘体的哈密顿量是这样实现的: 构建模型中的第三维度z映射时间。变换到旋转框架,波导螺旋周期需足够小,以实现z方向的绝热性(不懂,过完量力的绝热近似再说)。根据紧束缚方法,得到电场的动力学演化(看完紧束缚模型再说)。由动力学演化,用海森堡方程,或者量子郎之万方程反向变回去(文章上说用Peierls substitution,不过我大致估算了一下,海森堡方程就能得到相同结果。)就可以得到Hamiltonian。

该Hamiltonian含 周 期 性 变 化 的 第 三 维 度 , 使 用Floquet operator method和Fourier expansion method进行处理后按套路计算即可。能谱图还在画,争取今天画出来。

2024/06/19

感觉无心学习,稍微提前回去休息了。18,19号休息 周六日补上。

2024/06/18

17号弄到太晚, 18号一直很困就回去休息了.

2024/06/17

重复comsol案例:光子晶体和光子晶体带隙分析。注意光子晶体案例解出的电场是有虚部的,所以最后的线图直接用默认的normE和二维图对不上。comsol默认Ez显示实部解,因此ewfd.normE对应的不是sqrt(ewfd.Ex^2 + ewfd.E^2 + ewfd.Ez^2),而是sqrt(real(ewfd.Ex)^2 + imag(ewfd.Ex)^2 + real(ewfd.Ey)^2 + imag(ewfd.Ez)^2 + imag(ewfd.Ez)^2)。

2024/06/15 & 2024/06/16

休息。

This week

2024/06/14

看完Griffiths量子力学第四章。看高量课程,补画了个别图片。

2024/06/13

看Griffiths量子力学第四章。听了严以京关于开放系统中量子力学的报告。

2024/06/12

看Griffiths量子力学第四章。

2024/06/11

完成Griffiths量子力学第三章,读部分第四章。

2024/06/10

下午: 读Griffiths量子力学§3.1和§3.2并完成重点*习题。

2024/06/08 & 2024/06/09

从8号中午看到9号早上。

总结:完成习题1.3,浏览复现文章Ref. [5](Large Winding number)Winding number图片(其他图片套路和之前基本一致,就没画)。关于comsol仿真部分复现失败,对仿真掌握的知识太少,学过后可以回来尝试复现一下。

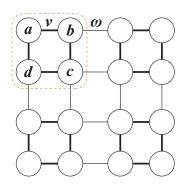


FIG. 1. 二维SSH模型。

习题1.3 A possible generalization to two dimensions-Consider a two dimensional generalization of the SSH model. Take parallel copies of the SSH chain and couple them without breaking chiral symmetry. What will happen with the edge states?

答: 习题1.3的答案有人在2017年发过prl(该书是2016年出版的) [6]。设二维SSH模型cell中子晶格顺时针标号为abcd, x(y)方向cell标号为i(j),如FIG. 1所示, Hamiltonian为:

$$\begin{split} \hat{H} = & v \sum_{i,j}^{N_{i},N_{j}} (\hat{a}_{i,j}^{\dagger} \hat{b}_{i,j} + \hat{b}_{i,j}^{\dagger} \hat{c}_{i,j} + \hat{c}_{i,j}^{\dagger} \hat{d}_{i,j} + \hat{d}_{i,j}^{\dagger} \hat{a}_{i,j}) \\ & + w \sum_{i,j}^{N_{i-1},N_{j}} (\hat{b}_{i,j}^{\dagger} \hat{a}_{i+1,j} + \hat{c}_{i,j}^{\dagger} \hat{d}_{i+1,j}) \\ & + w \sum_{i,j}^{N_{i},N_{j-1}} (\hat{d}_{i,j}^{\dagger} \hat{a}_{i,j+1} + \hat{c}_{i,j}^{\dagger} \hat{b}_{i,j+1}) + \text{H.c...} \end{split}$$

动量空间的Hamiltonian:

$$\hat{H}(k) = \begin{pmatrix} 0 & v + we^{-ik_x} & 0 & v + we^{-ik_y} \\ v + we^{ik_x} & 0 & v + we^{-ik_y} & 0 \\ 0 & v + we^{ik_y} & 0 & v + we^{-ik_x} \\ v + we^{ik_y} & 0 & v + we^{ik_x} & 0 \end{pmatrix}.$$

先画个色散关系,如FIG.2所示。

按Ref. [7]第一章的思路继续写不出了,因为不知道如何把 $\hat{H}(\mathbf{k})$ 写成实空间的 \hat{H} 来画能谱图以及波函数分布。

注意到Ref. [6]限制一个方向(y方向)无边界,另一个方向有界,利用该思想可以得到部分实空间

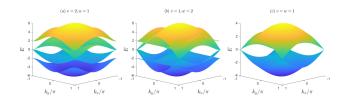


FIG. 2. 2D SSH模型色散关系。

的Hamiltonian:

$$\hat{H}(k) = \begin{pmatrix} A & B & 0 & \cdots & 0 \\ B^{\dagger} & A & B & \cdots & 0 \\ 0 & B^{\dagger} & A & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & B \\ 0 & 0 & 0 & \cdots & A \end{pmatrix}.$$

即主对角线是A矩阵,超对角线是B矩阵,次对角线是 B^{\dagger} 矩阵。其中

$$A = \begin{pmatrix} 0 & v & 0 & v + we^{-ik_y} \\ v & 0 & v + we^{-ik_y} & 0 \\ 0 & v + we^{ik_y} & 0 & v \\ v + we^{ik_y} & 0 & v & 0 \end{pmatrix},$$

$$B = \begin{pmatrix} 0 & 0 & 0 & 0 \\ w & 0 & 0 & 0 \\ 0 & 0 & 0 & w \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

可以据此画出能谱图,如FIG.3所示。原文似乎有些问题,跟据我所绘的图,蓝色能量对应的波函数并非边界态。

关于参考文献Ref. [6],过完基础知识之后回来仔细看 一看,了解物理图像。

下图是Ref. [5]的复现图,相比于我构想的winding number=2的模型,该模型动量空间的哈密顿量e指数符号不同,因此体现在实空间中的耦合项相差甚远。特别是,该模型还考虑的B模式与上一cell中A模式的耦合。但是在代码上几乎相同,此处不给出复现代码(用之前的调调参数,Hamiltonian改改符号即可)。Ref. [5]中Appendix A中动量空间Hamiltonian系数v1对应e指数少了个负号(复现图片会出现差异)。

能谱图如FIG。 5所示。但是看到文中标明4个边界态,同时就尝试在我的模型Winding number=2情况看一下,也出现了4个边界态,如FIG. 6所示。很神奇,后面继续学的时候关注一下相关知识。

2024/06/07

来了办公室但是超级困,周六日再来学吧。

2024/06/06

上午:睡觉(前两天熬的太晚)。

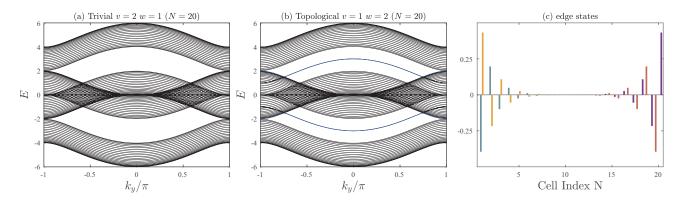


FIG. 3. 2D SSH模型能带(a)(b)以及边界态波函数分布(c)。

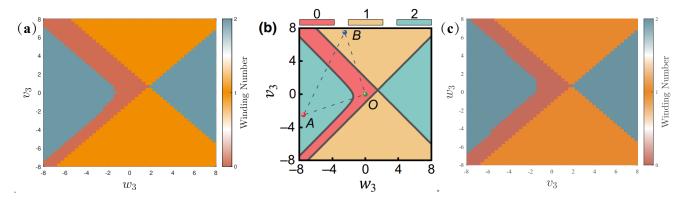


FIG. 4. Winding Number of Ref. [5], (a) 使用修正后Hamitonian。(b) 原图。(c) 使用原文Hamitonian。

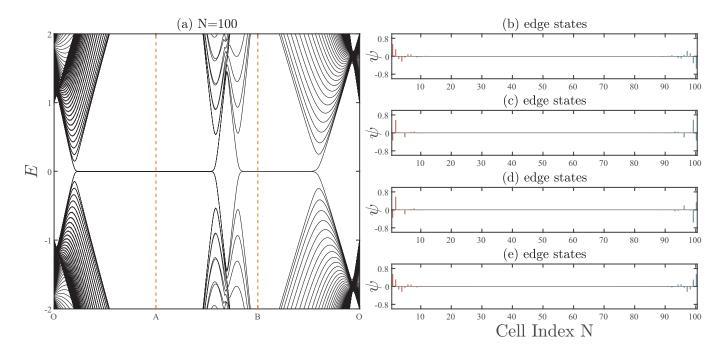


FIG. 5. Winding Number=2存在四个边界态,原文复现(原文N=120,但结果相同),2024/06/14补画

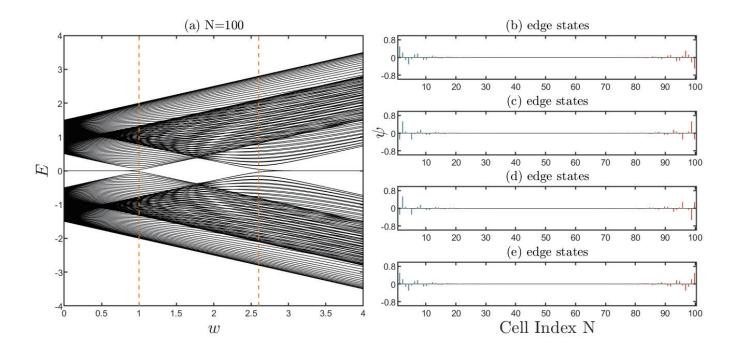


FIG. 6. Winding Number=2存在四个边界态

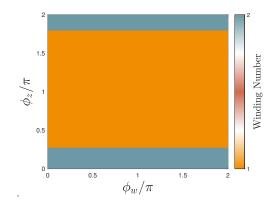


FIG. 7. 相位控制的Winding number。

下午:整理重绘复现图片,完成习题1.2(处理winding number图片变化交界处的方法不一样,所以三张图风格不统一,分别使用的是手动取消网格,光滑处理,光滑处理)。

习题1.2: Complex-valued hopping amplitudes- Generalize the SSH model in the following way. Assume that the hopping amplitudes $v=|v|e^{i\psi_v}$ and $w=|w|e^{i\psi_w}$ are complex, and include a third complex-valued hopping amplitude $z=|z|e^{i\psi_z}$ between the states $|m,A\rangle$ and $|m+1,B\rangle$ for every m. Provide a specific example where the tuning of one of the phases changes the bulk winding number.

答:和我设计的结构很像,但是远耦合方向相反且差个系数,并考虑相位。相位控制的Winding number如Fig. 7所示, v=0.5。w=0.5。代码如CODE4所示。

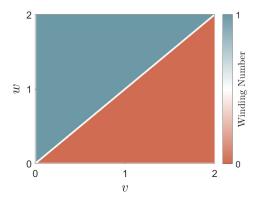


FIG. 8. SSH模型的Winding number。

2024/06/04 & 2024/06/05

读完A Short Course on Topological Insulators的第一章并复现图片 [7],如FIG. 8所示,代码位于CODE3。写了一个自动算2×2 Hamiltonian(频域)系统Winding number的函数,复现代码中可查。完成习题1.1并绘制图片。

习题1.1: Higher winding numbers- The SSH model is one-dimensional in space, and has a two-dimensional internal Hilbert space. Construct a lattice model that has these properties of the SSH model, but which has a bulk winding number of 2. Generalize the construction for an arbitrary integer bulk winding number.

答:如FIG.9所示。代码如CODE2所示。

$$\hat{H}(k) = \begin{pmatrix} 0 & v + e^{-ik} + 0.5we^{2ik} \\ v + e^{ik} + 0.5we^{-2ik} & 0 \end{pmatrix}.$$

2024/06/03

上午 & 下午: 完成Griffths量子力学第二章重点 (*) 习题。

2024/06/02

休息 & 读Griffths量子力学第二章。

2024/06/01

休息。

2024/05/31

摸鱼了。

2024/05/30

上午: 做学习记录。

下午:完成Griffiths量子力学第一章重点(*,**)习题。

2024/05/29

上午: 做Griffiths量子力学 $P_{1.1} - P_{1.7}$ 习题。

下午: 读A Short Course on Topological Insulators的§ 1.1-§ 1.3并复现如FIG. 10所示(直积不懂不会算,返回复习基础看过Griffiths第三章再往下阅读复现) [7]。MATLAB代码如CODE1所示(B、C、DE)。

2024/05/28

上午:看高量课程,复习经典力学。

下午:阅读Griffiths量子力学。

2024/05/27

重新制定基础知识学习计划,如FIG. 11所示。图书馆借相关书籍,看高量课程。

2024/05/25 & 2024/05/26

周六日休息。

2024/05/24

收拾房间,初步了解金属自由电子气体模型。

2024/05/23

办理校园卡,登记入住宿舍,收拾房间,选择工位。

- [1] C. Wang, et al., Realization of fractional quantum Hall state with interacting photons, Science 384, 579 (2024), https://www.science.org/doi/pdf/10.1126/science.ado3912.
- [2] H. ZHONG, D. MIHALACHE, S. SHEN, and Y. ZHANG, THE BAND STRUCTURE OF HELICAL WAVEGUIDE ARRAYS IN TOPOLOGICAL PHOTONICS: A TUTORIAL, .
- [3] M. C. Rechtsman, et al., Photonic Floquet topological insulators, Nature 496, 196 (2013).
- [4] F. Lederer, et al., Discrete solitons in optics, Physics Reports 463, 1 (2008).
- [5] H. Liu, et al., Acoustic Topological Metamaterials of Large Winding Number, Phys. Rev. Appl. 19, 054028 (2023).
- [6] F. Liu and K. Wakabayashi, Novel Topological Phase with a Zero Berry Curvature, Phys. Rev. Lett. 118, 076803 (2017).
- [7] J. K. Asb'th, L. Oroszl'ny, and A. P'lyi, A Short Course on Topological Insulators (Springer International Publishing, 2016).

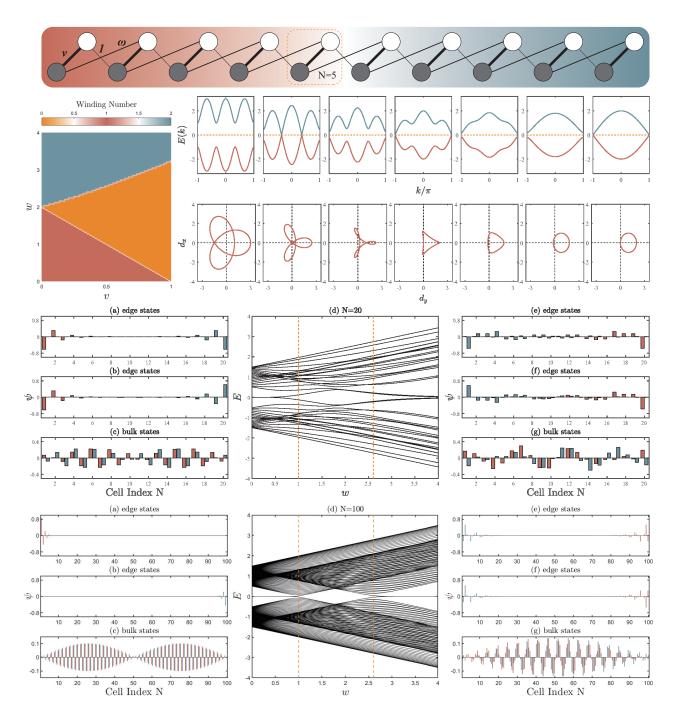
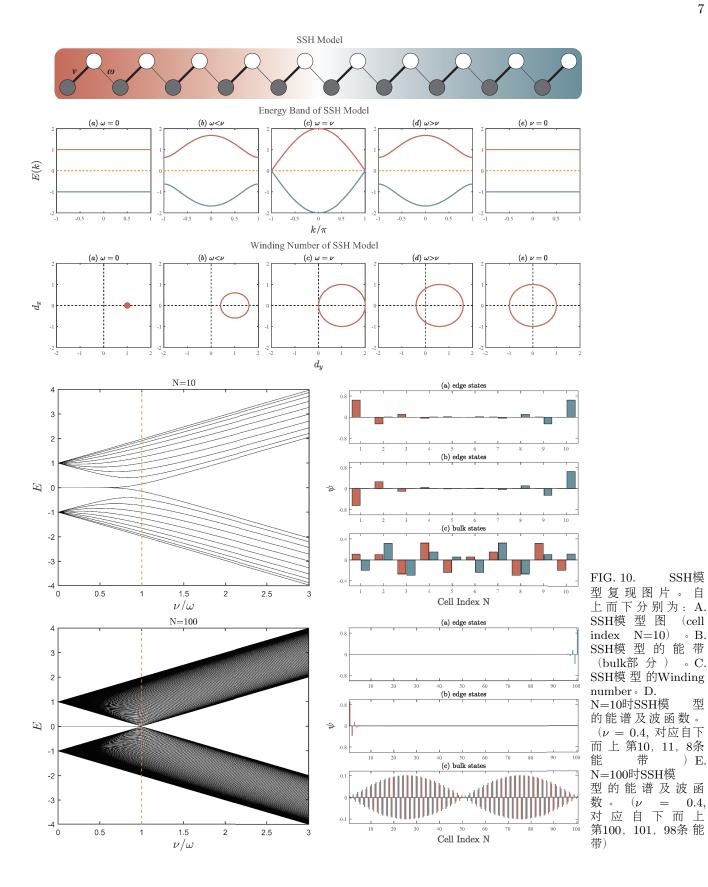


FIG. 9. 设计的Winding number=2的拓扑绝缘体。(Hypocycloid,圆内螺线)

```
2 clc
3 clear all
4
5 syms k;
6
7 rate=30;
8 parfor_progress(rate);
9
10 n=0;
11 vj=linspace(0,0*pi,rate);
```



```
wj = linspace(0, 2*pi, rate);
   z_j = linspace(0, 2*pi, rate);
13
   v0 = linspace (0.5, 0.5, rate);
   w0=linspace(1,1,rate);
15
   z0=linspace(1,1,rate);
16
17
   w=1;
18
19
   parfor a = 1
20
        pause (rand); % Replace with real code
21
22
        parfor_progress;
                 v=v0(a)*exp(1i*vj(a));
23
        for b = 1: rate
24
25
            w=w0(b)*exp(1i*wj(b));
26
            z=z0(b)*exp(1i*zj(b));
27
            H_k = [0 \text{ v+w*exp}(-1i*k)+z*exp(-2i*k); conj(v)+conj(w)*exp(1i*k)+conj(z)*
28
                \exp(2i*k) 0;
             [winding_number, d, d_0] = calculateWindingNumber(H_k);
29
            wn(a,b) = abs(winding_number);
30
            if abs(wn(a,b)) \le 1e-3 \mid (abs(wn(a,b))-1) \le 1e-3 \mid (abs(wn(a,b))-2)
31
                <= 1e-3
            else
32
                 \operatorname{wn}(a,b) = \operatorname{NaN};
33
            end
34
        \quad \text{end} \quad
35
   end
36
   parfor_progress(0);
37
   toc
38
39
   color0 = [1, 1, 1];
40
   color2 = [207, 108, 82] / 255;
41
   color1 = [241, 141, 0]/255;
   color3 = [109, 152, 165] / 255;
43
44
   Map = [linspace(color1(1), color0(1), 128)', linspace(color1(2), color0(2), 128)]
45
       ', linspace(color1(3), color0(3), 128)';
        linspace (color 0 (1), color 2 (1), 128), linspace (color 0 (2), color 2 (2), 128),
46
            linspace (color 0 (3), color 2 (3), 128);
        linspace (color2(1), color0(1), 128)', linspace (color2(2), color0(2), 128)',
47
            linspace (color 2 (3), color 0 (3), 128);
        linspace (color 0 (1), color 3 (1), 128), linspace (color 0 (2), color 3 (2), 128),
48
            linspace (color0(3), color3(3), 128)'];
49
50
   pcolor(wj/(pi), zj/(pi), abs(wn)')
51
   xlabel('$\phi_{w}\/pi$', 'interpreter', 'latex', 'FontSize', 22)
52
   ylabel('$\phi_{z}/\pi$', 'interpreter', 'latex', 'FontSize',22)
   \% axis ([0 2 0 2]
54
   c = colorbar('Ticks', [1 2], 'TickLabels', {'1', '2'}); ;%('ticks', linspace(-3,21,5),
       FontSize',11);
   c. Label. FontSize = 16;
56
   c.Label.Interpreter='Latex';
57
   c. Label. String = '$\mathrm{Winding\-Number}$';
   c. Label. Horizontal Alignment = "center";
   colormap (Map);
```

```
61
        delete (gcp);
 62
 63
 64
 65
        function [winding_number, d, d_0] = calculateWindingNumber(H_k)
 66
        syms k real;
 67
 68
        sigma_0 = [1 \ 0; \ 0 \ 1];
        sigma_x = [0 \ 1; \ 1 \ 0];
 70
        sigma_v = [0 -1i; 1i 0];
 71
        sigma_z = [1 \ 0; \ 0 \ -1];
 72
 73
        A = [reshape(sigma_0.', [], 1), reshape(sigma_x.', [], 1), reshape(sigma_y.', [], 1), reshape(sigma_
 74
                   1), reshape(sigma_z.', [], 1)];
        b = reshape(H_k, ', [], 1);
 75
 76
         coefficients = A \setminus b;
 77
 78
        d_0 = coefficients(1);
 79
        d_x = coefficients(2);
 80
        d_y = coefficients(3);
 81
        d_z = coefficients(4);
 82
 83
        d = [d_x; d_y; d_z];
 84
        d = simplify(d);
 85
 86
        d_{-}dk = diff(d, k);
 87
 88
        cross\_product = simplify(cross(d, d\_dk));
 89
        norm_d_squared = simplify(norm(d)^2);
 90
        integrand = cross_product(3) / norm_d_squared;
 91
 92
        func = matlabFunction(integrand, 'Vars', {k});
 93
 94
        winding_number = integral(func, -pi, pi) / (2 * pi);
 95
        end
 96
 97
       %%
        function percent = parfor_progress(N)
 99
        %PARFOR_PROGRESS Progress monitor (progress bar) that works with parfor.
100
                 PARFOR_PROGRESS works by creating a file called parfor_progress.txt in
101
                  your working directory, and then keeping track of the parfor loop's
                  progress within that file. This workaround is necessary because parfor
103
                  workers cannot communicate with one another so there is no simple way
                  to know which iterations have finished and which haven't.
105
106
                 PARFOR PROGRESS(N) initializes the progress monitor for a set of N
107
                  upcoming calculations.
108
109
                 PARFOR PROGRESS updates the progress inside your parfor loop and
110
                  displays an updated progress bar.
111
112
                 PARFOR_PROGRESS(0) deletes parfor_progress.txt and finalizes progress
113
114
115
```

```
To suppress output from any of these functions, just ask for a return
116
        variable from the function calls, like PERCENT = PARFOR_PROGRESS which
117
        returns the percentage of completion.
118
119
        Example:
120
121
           N = 100;
122
           parfor_progress(N);
123
           parfor i=1:N
124
               pause (rand); % Replace with real code
125
126
               parfor_progress;
127
           parfor_progress(0);
128
129
        See also PARFOR.
130
131
   % By Jeremy Scheff - jdscheff@gmail.com - http://www.jeremyscheff.com/
132
133
   error(nargchk(0, 1, nargin, 'struct'));
134
135
   if nargin < 1
136
        N = -1;
137
   end
138
139
   percent = 0;
140
   w = 50; % Width of progress bar
141
142
    if N > 0
143
        f = fopen('parfor_progress.txt', 'w');
144
        if f < 0
145
            error ('Do-you-have-write-permissions-for-%s?', pwd);
146
        end
147
        fprintf(f, '%d\n', N); % Save N at the top of progress.txt
148
        fclose(f);
149
150
        if nargout == 0
151
            disp(['--0%[>', repmat('-', 1, w), ']']);
152
        end
153
    elseif N = 0
154
        delete('parfor_progress.txt');
155
        percent = 100;
156
157
        if nargout == 0
158
            disp ([repmat(char(8), 1, (w+9)), char(10), '100%[', repmat('=', 1, w+1),
159
                ']']);
        end
160
    else
161
        if ~exist('parfor_progress.txt', 'file')
162
            error('parfor_progress.txt-not-found.-Run-PARFOR_PROGRESS(N)-before-
163
                PARFOR_PROGRESS-to-initialize-parfor_progress.txt.');
        end
164
165
        f = fopen('parfor_progress.txt', 'a');
166
        fprintf(f, '1\n');
167
        fclose(f);
168
169
```

```
f = fopen('parfor_progress.txt', 'r');
170
        progress = fscanf(f, '%d');
171
        fclose(f);
172
        percent = (length(progress)-1)/progress(1)*100;
173
174
        if nargout = 0
175
             perc = sprintf('%3.0f\%', percent); % 4 characters wide, percentage
176
             disp([repmat(char(8), 1, (w+9)), char(10), perc, '[', repmat('=', 1,
177
                round (percent *w/100)), ^{\prime}>, repmat(^{\prime}-, 1, w - round (percent *w/100)),
                ']']);
178
        end
   end
179
```

```
tic
   clc
2
   clear all
   syms k;
5
   rate = 20;
7
   % parfor_progress(rate);
   n=0;
   va = linspace(0, 2, rate+1);
11
   va(1) = [];
   wb = linspace(0, 2, rate+1);
13
   wb(1) = [];
14
   wn = NaN(rate, rate);
15
16
   parpool;
^{17}
   parfor a = 1: rate
18
        % pause(rand); % Replace with real code
19
        % parfor_progress;
20
        v=va(a);
21
        for b = 1: rate
22
             w=wb(b);
23
             H_k = [0 \ v + w*exp(-1i*k); \ v + w*exp(1i*k) \ 0];
24
             [winding_number, d, d_0] = calculateWindingNumber(H_k);
25
             if v == w
26
                  \operatorname{wn}(a,b) = \operatorname{NaN};
27
             else
28
                  wn(a,b) = winding\_number;
29
             end
30
        \quad \text{end} \quad
31
   end
32
   delete (gcp);
33
   % parfor_progress(0);
34
35
36
   color1 = [207, 108, 82] / 255;
37
   color 2 = [1, 1, 1];
```

```
color3 = [109, 152, 165] / 255;
40
   Map = [linspace(color1(1), color2(1), 128)', linspace(color1(2), color2(2), 128)]
41
        , linspace(color1(3), color2(3), 128);
       linspace (color2(1), color3(1), 128), linspace (color2(2), color3(2), 128),
42
           linspace (color 2 (3), color 3 (3), 128);
43
   %%
44
   pcolor (va, wb, wn')
   shading interp
46
   set(gca, 'XTick', [0,1,2], 'FontSize',16)
set(gca, 'YTick', [0,1,2], 'FontSize',16)
48
   xlabel ('$v$', 'interpreter', 'latex', 'FontSize', 22)
49
   ylabel ('$w$', 'interpreter', 'latex', 'FontSize', 22)
50
   axis ([0 2 0 2])
51
   c = colorbar('Ticks', [0\ 1]);;%('ticks', linspace(-3,21,5), 'FontSize',11);
52
   c. Label. FontSize=16;
53
   c. Label. Interpreter='Latex';
54
   c. Label. String = '$\mathrm{Winding\-Number}$';
55
   c.Label.HorizontalAlignment = "center";
   colormap (Map);
57
   %save("Phase_diagram_winding_number",'-mat')
59
   function [winding_number, d, d_0] = calculateWindingNumber(H_k)
60
   syms k real;
61
62
   sigma_0 = [1 \ 0; \ 0 \ 1];
63
   sigma_x = [0 \ 1; \ 1 \ 0];
64
   sigma_v = [0 -1i; 1i 0];
65
   sigma_z = [1 \ 0; \ 0 \ -1];
66
67
   A = [reshape(sigma_0.', [], 1), reshape(sigma_x.', [], 1), reshape(sigma_y.', [], 1)]
68
        1), reshape(sigma_z.', [], 1)];
   b = reshape(H_k.', [], 1);
69
70
   coefficients = A \setminus b;
71
   d_0 = coefficients(1);
73
   d_x = coefficients(2);
   d_{-v} = coefficients(3);
75
   d_z = coefficients(4);
76
77
   d = [d_x; d_y; d_z];
78
   d = simplify(d);
79
80
   d_{-}dk = diff(d, k);
81
82
   cross_product = simplify(cross(d, d_dk));
83
   norm_d_squared = simplify(norm(d)^2);
84
   integrand = cross_product(3) / norm_d_squared;
85
86
   func = matlabFunction(integrand, 'Vars', {k});
87
88
   winding_number = integral(func, -pi, pi) / (2 * pi);
89
90
```

```
tic
2
   clc
   clear all
   syms k;
   rate=20;
   parfor_progress(rate);
10
   va = linspace(0.0001, 1, rate);
11
   wb = linspace(0.0001, 4, rate);
12
   wn = NaN(rate, rate);
13
14
   parfor a = 1: rate
15
        parfor_progress;
16
        v=va(a);
17
        for b = 1: rate
18
            w=wb(b);
19
            H_k = [0 \text{ v} + 1()*\exp(-1i*k) + 0.5*w*\exp(1i*2*k); \text{ v} + 1*\exp(1i*k) + 0.5*w*\exp(-1i*k)]
20
                (-1i*2*k) 0;
            [winding_number, d, d_0] = calculateWindingNumber(H_k);
21
            wn(a,b) = abs(winding_number);
22
            if abs(wn(a,b)) \le 1e-3 \mid | (abs(wn(a,b))-1) \le 1e-3 \mid | (abs(wn(a,b))-2)
23
                <= 1e-3
            else
24
                 \operatorname{wn}(a,b) = \operatorname{NaN};
25
            end
26
        end
27
28
   end
   parfor_progress(0);
29
   toc
30
31
   color0 = [1, 1, 1];
32
   color2 = [207, 108, 82] / 255;
33
   color1 = [241, 141, 0]/255;
34
   color3 = [109, 152, 165] / 255;
35
36
   Map = [linspace(color1(1), color0(1), 128)', linspace(color1(2), color0(2), 128)]
37
       ', linspace (color1(3), color0(3), 128)';
        linspace (color 0 (1), color 2 (1), 128), linspace (color 0 (2), color 2 (2), 128),
38
            linspace(color0(3), color2(3), 128);
        linspace (color2(1), color0(1), 128), linspace (color2(2), color0(2), 128),
39
            linspace (color2(3), color0(3), 128);
        linspace (color 0 (1), color 3 (1), 128), linspace (color 0 (2), color 3 (2), 128),
40
            linspace (color 0 (3), color 3 (3), 128) '];
42
   pcolor (va, wb, abs (wn))
   shading interp
   \% set(gca, 'XTick', [0,1,2], 'FontSize', 16)
   % set(gca, 'YTick', [0,1,2], 'FontSize', 16)
   xlabel('$v$', 'interpreter', 'latex', 'FontSize', 22)
```

```
ylabel ('$w$', 'interpreter', 'latex', 'FontSize', 22)
       \% axis ([0 2 0 2])
 49
       \mathbf{c} = \mathbf{colorbar}; \%('ticks', linspace(-3,21,5), 'FontSize', 11);
       c. Label. FontSize=16;
 51
       c.Label.Interpreter='Latex';
       c. Label. String = '$\mathrm{Winding\-Number}$';
 53
       c.Label.HorizontalAlignment = "center";
 54
       colormap (Map);
 55
       \% save('WN_012.mat')
       delete (gcp);
 57
 58
 59
       function [winding_number, d, d_0] = calculateWindingNumber(H_k)
 60
       syms k real;
 61
 62
       sigma_0 = [1 \ 0; \ 0 \ 1];
 63
       sigma_x = [0 \ 1; \ 1 \ 0];
 64
       sigma_y = [0 -1i; 1i 0];
 65
       sigma_z = [1 \ 0; \ 0 \ -1];
 66
 67
       A = [reshape(sigma_0.', [], 1), reshape(sigma_x.', [], 1), reshape(sigma_y.', [], 1), reshape(sigma_
 68
                  1), reshape(sigma_z.', [], 1)];
       b = reshape(H_k, ', [], 1);
 69
 70
        coefficients = A \setminus b;
 71
 72
       d_0 = coefficients(1);
 73
       d_x = coefficients(2);
 74
       d_{-y} = coefficients(3);
 75
       d_z = coefficients(4);
 76
 77
       d = [d_x; d_y; d_z];
 78
       d = simplify(d);
 79
 80
       d_{-}dk = diff(d, k);
 81
 82
        cross\_product = simplify(cross(d, d\_dk));
 83
       norm_d_squared = simplify(norm(d)^2);
 84
       integrand = cross_product(3) / norm_d_squared;
 86
       func = matlabFunction(integrand, 'Vars', {k});
 87
 88
       winding_number = integral(func, -pi, pi) / (2 * pi);
 89
       end
 90
 91
 92
       function percent = parfor_progress(N)
 93
       %PARFOR_PROGRESS Progress monitor (progress bar) that works with parfor.
 94
                PARFOR PROGRESS works by creating a file called parfor progress.txt in
 95
                 your working directory, and then keeping track of the parfor loop's
 96
                 progress within that file. This workaround is necessary because parfor
 97
                 workers cannot communicate with one another so there is no simple way
 98
                 to know which iterations have finished and which haven't.
 99
100
                PARFOR PROGRESS(N) initializes the progress monitor for a set of N
101
                 upcoming calculations.
102
```

```
103
        PARFOR PROGRESS updates the progress inside your parfor loop and
104
        displays an updated progress bar.
105
106
        PARFOR_PROGRESS(0) deletes parfor_progress.txt and finalizes progress
107
108
        To suppress output from any of these functions, just ask for a return
110
        variable from the function calls, like PERCENT = PARFOR_PROGRESS which
        returns the percentage of completion.
112
        Example:
114
115
           N = 100;
116
           parfor_progress(N);
117
           parfor i=1:N
118
               pause (rand); % Replace with real code
119
               parfor_progress;
120
121
           parfor_progress(0);
122
123
        See also PARFOR.
124
125
   % By Jeremy Scheff - jdscheff@gmail.com - http://www.jeremyscheff.com/
126
127
   error(nargchk(0, 1, nargin, 'struct'));
128
129
    if nargin < 1
130
        N = -1:
131
   end
132
133
   percent = 0;
134
   w = 50; % Width of progress bar
135
136
    if N > 0
137
        f = fopen('parfor_progress.txt', 'w');
138
        if f < 0
139
             error ('Do-you-have-write-permissions-for-%s?', pwd);
140
        end
141
        fprintf(f, '%d\n', N); % Save N at the top of progress.txt
142
        fclose(f);
143
144
        if nargout == 0
145
             disp(['--0%[>', repmat('-', 1, w), ']']);
146
        end
    elseif N == 0
148
        delete('parfor_progress.txt');
149
        percent = 100;
150
151
        if nargout == 0
152
             disp ([repmat(char(8), 1, (w+9)), char(10), '100%[', repmat('=', 1, w+1),
153
                ']']);
        end
154
    else
155
        if ~exist('parfor_progress.txt', 'file')
156
```

```
error('parfor_progress.txt-not-found.-Run-PARFOR_PROGRESS(N)-before-
157
                PARFOR_PROGRESS-to-initialize-parfor_progress.txt.');
        end
158
159
        f = fopen('parfor_progress.txt', 'a');
160
        fprintf(f, '1\n');
161
        fclose(f);
162
163
        f = fopen('parfor_progress.txt', 'r');
164
        progress = fscanf(f, '%d');
165
        fclose(f);
166
        percent = (length(progress)-1)/progress(1)*100;
167
168
        if nargout == 0
169
            perc = sprintf('%3.0f\%'', percent); \% 4 characters wide, percentage
170
            disp ([repmat(char(8), 1, (w+9)), char(10), perc, '[', repmat('=', 1,
171
                round(percent*w/100)), '>', repmat('\cdot, ', 1, w - round(percent*w/100)),
                ']']);
        end
172
   end
173
```

```
clear all
              clc
  2
              f1 = figure(1);
              position0=get(f1, 'position');
              set (f1, 'position', position0+[-0.8*position0(3), 0, 1.6*position0(3), -0.3*position0
                               (4)]):
              t=tiledlayout ("horizontal", "TileSpacing", "tight");
               titlej = \{ (a) \land omega = 0\$; (b) \land omega \land textless \land u\$; (c) \land omega = \land u\$; (d) \land 
                              \langle nega \rangle textgreater nu\$'; '\$(e) \rangle -nu=0\$';
              rate_k = 1000;
  9
10
              vj = [0, 0.3, 0.5, 0.6, 0.7, 0.8, 1];
11
              wj = [4, 2.3, 1.5, 0.8, 0.3, 0, 0];
 ^{12}
              ki=linspace(-pi, pi, rate_k);
13
14
              for j = 1 : 7
15
                                 v=vj(j);
 16
                                w=wj(j);
17
                                 for i = 1 : rate_k
18
                                                   k=ki(i);
 19
                                                   H_k = [0 \text{ v} + 1*\exp(-1i*k) + 0.5*w*\exp(1i*2*k); \text{ v} + 1*\exp(1i*k) + 0.5*w*\exp(-1i*k)]
20
                                                                    i*2*k) 0;
                                                     [d, d_0] = WindingNumberFigure(H_k);
21
                                                    d_x(i) = d(1);
 22
                                                    d_y(i) = d(2);
23
                                 end
24
                                 nexttile
25
                                 plot (linspace (-3.6, 3.6, 100), linspace (0, 0, 100), 'LineWidth', 1, 'linestyle', '—',
26
                                                   'Color', 'k')
                                 hold on
27
                                 \operatorname{plot}(\operatorname{linspace}(0,0,100),\operatorname{linspace}(-4,4,100),\operatorname{'LineWidth'},1,\operatorname{'linestyle'},\cdot
28
                                                  Color ', 'k')
                                 hold on
29
```

```
plot (d_x, d_y, 'LineWidth', 1.5, 'Color', [207, 108, 82]/255);
30
                       hold on
31
                       set (gca , 'LineWidth' ,1);
32
                       axis([-3.6,3.6,-4,4])
33
                       set (gca, 'ytick', [-4 -2 0 2 4], 'FontSize', 12);
34
                       set(gca, 'xtick', [-3 \ 0 \ 3], 'FontSize', 12);
35
                      % title(titlej{j}, 'interpreter', 'latex', 'FontSize', 12);
36
         end
37
         % title(t,'Winding Number of SSH Model','FontSize',18);
         xlabel(t, '$d_{y}$', 'interpreter', 'latex', 'FontSize', 18);
39
          ylabel(t, '$d_{x}$', 'interpreter', 'latex', 'FontSize', 18);
40
41
          function [d, d<sub>0</sub>] = WindingNumberFigure(H<sub>k</sub>)
42
         syms k v w
43
44
         sigma_0 = [1 \ 0; \ 0 \ 1];
45
         sigma_x = [0 \ 1; \ 1 \ 0];
46
         sigma_y = [0 -1i; 1i 0];
47
         sigma_z = [1 \ 0; \ 0 \ -1];
48
49
         A = [reshape(sigma_0.', [], 1), reshape(sigma_x.', [], 1), reshape(sigma_y.', [], 1), reshape(sigma_
50
                        1), reshape(sigma_z.', [], 1)];
         b = reshape(H_k, ', [], 1);
51
52
         coefficients = A \setminus b;
53
54
         d_0 = coefficients(1);
55
         d_x = coefficients(2);
56
         d_{-y} = coefficients(3);
57
         d_z = coefficients(4);
58
59
         d = [d_x; d_y; d_z];
60
         \%d = simplify(d);
61
         end
62
```

```
clear all
   clc
2
3
   f2 = figure(1);
   position0=get(f2, 'position');
   set (f2, 'position', position0+[-0.85*position0(3), -0.3*position0(4), 1.7*position0
       (3), 0*position 0 (4) ]);
   t=tiledlayout (3,3," TileSpacing"," tight");
   f2 = figure(1);
   position0=get(f2, 'position');
11
   set (f2, 'position', position0+[-0*position0(3), -0*position0(4), -0.2*position0(3)]
12
       ,0.34*position0(4)]);
13
   n=100;
14
   %%
15
   rate_v = 500;
16
17
  v = 0.5;
18
```

```
wj = linspace(0, 4, rate_v);
   for j = 1: rate_v
20
        w=wj(j);
21
        v1 = [v, 1] * ones (1, n);
22
        v1=v1(:)';
23
        v1(2*n) = [];
24
        w0 = [0.5*w, 0] '* ones (1, n);
25
        w0=w0(:);
26
        w0(2*n) = [];
27
        w0(2*n-1) = [];
28
        w0(2*n-2) = [];
29
        w0(2*n-3) = [];
30
        w0(2*n-4) = [];
31
        H=diag(v1,1)+diag(v1,-1)+diag(w0,5)+diag(w0,-5);
32
        [V,D] = eig(H);
33
        E(j,:) = diag(D);
34
        \%psi(:,:,j)=V;
35
   end
36
   nexttile (2,[3,1]);
37
   for j = 1 : size(E,2)
38
        plot (wj, E(:, j), "Color", 'black');
39
        hold on;
40
   end
41
   plot (linspace (1,1,100), linspace (-4,4,100), 'LineWidth', 1, 'linestyle', '—', 'Color'
42
        ,[241,141,0]/255)
   hold on;
   plot (linspace (2.6, 2.6, 100), linspace (-4, 4, 100), 'LineWidth', 1, 'linestyle', '---', '
44
       Color', [241,141,0]/255)
   hold on;
45
   set (gca, 'LineWidth', 1);
46
   xlabel('$w$', 'interpreter', 'latex', 'FontSize',16);
47
   ylabel('$E$', 'interpreter', 'latex', 'FontSize', 16);
48
   title (sprintf ("(d) N=%d", n), 'interpreter', 'latex', 'FontSize', 12);
49
50
   %ratio = 400;
51
52
   v = 0.5;
53
   w=0;
54
   v1 = [v, 1] * ones (1, n);
   v1=v1(:)';
56
   v1(2*n) = [];
57
   w0 = [0.5*w, 0] '* ones (1, n);
58
   w0=w0(:);
   w0(2*n) = [];
60
   w0(2*n-1) = [];
61
   w0(2*n-2) = [];
62
   w0(2*n-3) = [];
63
   w0(2*n-4) = [];
64
   H=diag(v1,1)+diag(v1,-1)+diag(w0,5)+diag(w0,-5);
65
   [V,D] = eig(H);
66
   psi=V;
67
   x = linspace(0.75, 10.25, 20);
69
   nexttile(1);
70
   %bar(x, reshape(psi(:,10, ratio),[2,10]))
71
   b1=bar(reshape(psi(:,n),[2,n])',0.9, 'GroupWidth',0.8);
```

```
set(gca, 'ytick', [-0.8, 0, 0.8])
    axis([0.5, n+0.5, -1, 1])
74
    title ({ '(a) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
        FontSize',12);
    set (b1(1), 'FaceColor', [207, 108, 82]/255);
    set (b1(2), 'FaceColor', [109,152,165]/255);
77
    set (gca, 'LineWidth', 1);
78
79
    nexttile (4);
   %bar(x, reshape(psi(:,11, ratio),[2,10]))
81
    b2=bar (reshape (psi (:, n+1), [2, n])', 0.9, 'GroupWidth', 0.8);
    set(gca, 'ytick', [-0.8, 0, 0.8])
83
    axis([0.5, n+0.5, -1, 1])
84
    title ({ '(b) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
85
        FontSize', 12);
    set(b2(1), 'FaceColor', [207,108,82]/255);
    set (b2(2), 'FaceColor', [109,152,165]/255);
87
    set (gca , 'LineWidth' ,1);
    ylabel ('$\psi$', 'interpreter', 'latex', 'FontSize', 16, 'VerticalAlignment', 'middle')
89
90
    nexttile (7);
91
    %bar(x, reshape(psi(:, 8, ratio), [2, 10]))
92
    b3=bar(reshape(psi(:,n-2),[2,n])',0.9,'GroupWidth',0.8);
    if n >= 100
94
         set(gca, 'ytick', [-0.1, 0, 0.1])
95
         axis([0.5, n+0.5, -0.15, 0.15])
96
    else
97
         set(gca, 'ytick', [-0.4, 0, 0.4])
98
         axis([0.5, n+0.5, -0.5, 0.5])
99
    end
100
    title ({ '(c) bulk states'}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
101
        FontSize', 12);
    set (b3(1), 'FaceColor', [207,108,82]/255);
102
    \underline{\mathsf{set}}\,(\,\mathrm{b3}\,(2)\,\,,\,{}^{\backprime}\mathsf{FaceColor}\,\,{}^{\backprime}\,,[109\,,\!152\,,\!165]/255)\,;
103
    set (gca, 'LineWidth', 1);
104
    xlabel('Cell-Index-N', 'interpreter', 'latex', 'FontSize', 16);
106
107
108
109
    v = 0.5;
110
   w=4;
    v1 = [v, 1] * ones (1, n);
112
    v1=v1(:)';
    v1(2*n) = [];
114
   w0 = [0.5*w, 0] '* ones (1, n);
   w0=w0(:);
116
    w0(2*n) = [];
117
   w0(2*n-1) = [];
   w0(2*n-2) = [];
119
   w0(2*n-3) = [];
120
   w0(2*n-4) = [];
121
   H=diag(v1,1)+diag(v1,-1)+diag(w0,5)+diag(w0,-5);
   [V,D] = eig(H);
123
   psi=V;
124
```

```
125
    x = linspace(0.75, 10.25, 20);
126
    nexttile(3);
127
    %bar(x, reshape(psi(:,10, ratio),[2,10]))
128
    b1=bar(reshape(psi(:,n),[2,n])',0.9, 'GroupWidth',0.8);
    set(gca, 'ytick', [-0.8, 0, 0.8])
130
    axis([0.5, n+0.5, -1, 1])
131
    title ({ '(e) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
132
        FontSize', 12);
    set (b1(1), 'FaceColor', [207, 108, 82]/255);
133
    \underline{\mathsf{set}}\,(\,\mathrm{b1}\,(2)\,\,,\,{}^{\backprime}\mathsf{FaceColor}\,\,{}^{\backprime}\,,[109\,,\!152\,,\!165]/255)\,;
134
    set (gca , 'LineWidth' ,1);
135
136
    nexttile (6);
137
    %bar(x, reshape(psi(:,11, ratio),[2,10]))
138
    b2=bar (reshape (psi (:, n+1), [2, n])', 0.9, 'GroupWidth', 0.8);
    set(gca, 'ytick', [-0.8, 0, 0.8])
140
    axis([0.5, n+0.5, -1, 1])
141
    title ({ '(f) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
142
        FontSize',12);
    set (b2(1), 'FaceColor', [207,108,82]/255);
143
    set (b2(2), 'FaceColor', [109,152,165]/255);
    set (gca, 'LineWidth',1);
145
    ylabel ('$\psi$', 'interpreter', 'latex', 'FontSize', 16, 'VerticalAlignment', 'middle')
146
    nexttile (9);
148
    %bar(x, reshape(psi(:, 8, ratio), [2, 10]))
149
    b3=bar(reshape(psi(:,n-2),[2,n])',0.9, 'GroupWidth',0.8);
150
    if n >= 100
151
         set(gca, 'ytick', [-0.1, 0, 0.1])
152
         axis([0.5, n+0.5, -0.15, 0.15])
153
    else
154
         set(gca, 'ytick', [-0.4, 0, 0.4])
155
         axis([0.5, n+0.5, -0.5, 0.5])
156
    end
157
    title ({ '(g) bulk states'}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
        FontSize', 12);
    set (b3(1), 'FaceColor', [207, 108, 82]/255);
    set (b3(2), 'FaceColor', [109,152,165]/255);
160
    set (gca, 'LineWidth', 1);
161
162
    xlabel ('Cell-Index-N', 'interpreter', 'latex', 'FontSize', 16);
```

```
titlej = { `$(a) \land omega=0$ `; `$(b) \land omega \land extless \land u$ `; `$(c) \land omega= \land u$ `; `$(d) \land omega= \land u$ `; `$(
                           \langle nega \rangle textgreater nu\$'; '\$(e) \rangle nu=0\$';
            rate_k = 1000;
  9
10
            vj = [1, 1, 1, 0.6, 0];
11
            wj = [0, 0.6, 1, 1, 1];
12
            ki=linspace(-pi, pi, rate_k);
13
            for j = 1 : 5
15
                              v=vj(j);
16
                             w=wj(j);
17
                              for i = 1 : rate_k
18
                                              k=ki(i);
19
                                               E_{-1}(i) = sqrt(v+w+2*v*w*cos(k));
20
                                               E_{-2}(i) = -sqrt(v+w+2*v*w*cos(k));
21
                              end
22
                              nexttile
23
                              plot (ki/pi, linspace (0,0, rate_k), 'LineWidth', 1, 'linestyle', '—', 'Color'
24
                                              ,[241,141,0]/255)
                              hold on
25
                              plot (ki/pi, E<sub>-1</sub>, 'LineWidth', 1.5, 'Color', [207, 108, 82]/255);
26
                              hold on
27
                              plot (ki/pi, E<sub>2</sub>, 'LineWidth', 1.5, 'Color', [109, 152, 165]/255);
28
                              hold on
29
                              set(gca, 'LineWidth',1);
30
                              axis([-1,1,-2,2])
31
                               title (titlej { j }, 'interpreter', 'latex', 'FontSize', 12);
32
            end
33
             title (t, 'Energy-Band-of-SSH-Model', 'FontSize', 16);
34
            xlabel(t, '$k/\pi$', 'interpreter', 'latex', 'FontSize',16);
35
            ylabel(t, '$E(k)$', 'interpreter', 'latex', 'FontSize', 16);
```

```
clear all
                   clc
                 f1 = figure(1);
                   position0=get(f1, 'position');
                   set(f1, 'position', position0+[-0.75*position0(3), 0, 1.5*position0(3), -0.27*]
                                         position 0 (4) ]);
                   t=tiledlayout ("horizontal", "TileSpacing", "tight");
                    titlej = { (3) } \circ omega = 0 ; (3) } \circ omega + textless \setminus omega + (3) } \circ omega = (3) } \circ 
                                        \langle nega \rangle textgreater nu\$'; '\$(e) \rangle - nu=0\$';
                   rate_k = 1000;
   9
10
                   vj = [1, 1, 1, 0.6, 0];
 11
                   w_j = [0, 0.6, 1, 1, 1];
12
                   ki=linspace(-pi, pi, rate_k);
13
14
                   for j = 1 : 5
15
                                            v=vj(j);
16
                                          w=wj(j);
17
                                            for i = 1 : rate_k
18
                                                                     k=ki(i);
19
                                                                     d_x(i)=v+w*\cos(k);
20
```

```
d_y(i)=w*sin(k);
21
       end
22
       nexttile
23
       plot(linspace(-2,2,100), linspace(0,0,100), 'LineWidth', 1, 'linestyle', '---', '
24
           Color ', 'k')
       hold on
25
        plot(linspace(0,0,100), linspace(-2,2,100), 'LineWidth', 1, 'linestyle', '---', '
26
           Color ', 'k')
       hold on
       if i = 1
28
            plot (d_x,d_y,'ro','MarkerFaceColor',[207,108,82]/255,'MarkerSize',9);
29
30
            plot (d_x, d_y, 'LineWidth', 1.5, 'Color', [207, 108, 82]/255);
31
       end
32
       hold on
33
       set (gca, 'LineWidth', 1);
34
       axis([-2,2,-2,2])
35
        title(titlej{j}, 'interpreter', 'latex', 'FontSize',12);
36
37
   title (t, 'Winding Number of SSH Model', 'FontSize', 16);
   xlabel(t, '$d_{y}$', 'interpreter', 'latex', 'FontSize',16);
39
   ylabel(t, '$d_{x}$', 'interpreter', 'latex', 'FontSize',16);
```

```
clear all
1
   clc
2
   rate_{-}v = 30000;
   n = 100;
5
   vj = linspace(0, 3, rate_v);
   for j = 1: rate_v
10
        v=vj(j);
11
        vw = [v, w] * ones (1, n);
^{12}
        vw=vw(:) ';
13
        vw(2*n) = [];
14
        H=diag(vw,1)+diag(vw,-1);
15
        [V,D] = eig(H);
16
        E(j,:) = diag(D);
17
        \%psi(:,:,j)=V;
18
   end
19
   for j = 1 : size(E, 2)
20
             plot(vj,E(:,j),"Color",'black');
^{21}
             hold on;
22
   end
   plot (linspace (1,1,100), linspace (-4,4,100), 'LineWidth', 1, 'linestyle', '—', 'Color'
^{24}
        ,[241,141,0]/255)
   set (gca, 'LineWidth',1);
25
   {\tt xlabel('\$\nu/\omega\$', 'interpreter', 'latex', 'FontSize', 16);}
26
   ylabel('$E$', 'interpreter', 'latex', 'FontSize', 16);
27
   title (sprintf ("N=%d", n), 'interpreter', 'latex', 'FontSize', 12);
28
29
30
   f2 = figure(2);
```

```
position0=get(f2, 'position');
   set (f2, 'position', position0+[-0.4*position0(3), -0.3*position0(4), 0.2*position0(3)]
33
       ,0.3*position0(4)]);
   t=tiledlayout ("vertical", "TileSpacing", "tight");
34
   %ratio = 400:
36
37
   v = 0.4;
38
   w=1;
   vw = [v, w] * ones (1, n);
40
   vw=vw(:) ';
   vw(2*n) = [];
42
  H=diag(vw,1)+diag(vw,-1);
43
   [V,D] = eig(H);
44
   psi=V;
45
46
   x = linspace(0.75, 10.25, 20);
47
   nexttile
   %bar(x, reshape(psi(:,10, ratio),[2,10]))
49
   b1=bar(reshape(psi(:,n),[2,n])',0.9, 'GroupWidth',0.8);
   set(gca, 'ytick', [-0.8, 0, 0.8])
51
   axis([0.5, n+0.5, -1, 1])
   title ({ '(a) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
53
       FontSize', 12);
   set (b1(1), 'FaceColor', [207,108,82]/255);
54
   set (b1(2), 'FaceColor', [109,152,165]/255);
55
56
   nexttile
57
   %bar(x, reshape(psi(:,11, ratio),[2,10]))
58
   b2=bar (reshape (psi (:, n+1), [2, n])', 0.9, 'GroupWidth', 0.8);
59
   set(gca, 'ytick', [-0.8, 0, 0.8])
60
   axis([0.5, n+0.5, -1, 1])
61
   title ({ '(b) - edge - states '}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
       FontSize', 12);
   set (b2(1), 'FaceColor', [207,108,82]/255);
   set (b2(2), 'FaceColor', [109,152,165]/255);
64
66
   %bar(x, reshape(psi(:, 8, ratio), [2, 10]))
   b3=bar(reshape(psi(:,n-2),[2,n])',0.9,'GroupWidth',0.8);
68
   if n >= 100
69
       set(gca, 'ytick', [-0.1, 0, 0.1])
70
       axis([0.5, n+0.5, -0.12, 0.12])
71
   else
72
        set(gca, 'ytick', [-0.4, 0, 0.4])
73
   axis ([0.5, n+0.5, -0.5, 0.5])
74
75
   title ({ '(c) bulk states'}, 'Vertical Alignment', 'bottom', 'interpreter', 'latex', '
76
       FontSize', 12);
   set (b3(1), 'FaceColor', [207,108,82]/255);
77
   set (b3(2), 'FaceColor', [109,152,165]/255);
78
   ylabel(t, '$\psi$', 'interpreter', 'latex', 'FontSize', 16);
80
   xlabel(t, 'Cell-Index-N', 'interpreter', 'latex', 'FontSize', 16);
```

学习规划

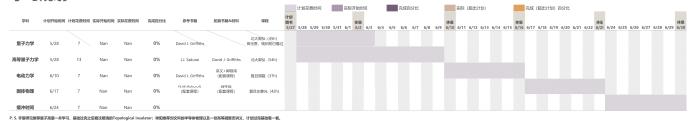


FIG. 11. 基础知识学习计划