Appendices

Appendix A

Sample Matlab Code to Illustrate Ill-Conditioning

```
%Simple Matlab code to illustrate ill-conditioning
%with a numerical example
%-------------------
%
%Set Parameters
dim = 5; %sets dimension of the matrix
numiter = 10000; %number of iterations to run through
noiselevel =1e-3; %scales the magnitude of the random noise vector
%
%--------------------
%
%Define the matrix and input vector
A=1./([1:dim]'*ones(1,dim)+ones(dim,1)*[0:1:dim-1]);
    %This defines the Hilbert matrix of size dimXdim
xin = ones(dim,1);
xout = A*xin;
%
%-----------------
%
```

```
%Statistics for the 'forward problem'
for kk = 1:numiter
    noisein = noiselevel*randn(dim,1);
    ein = norm(noisein)./norm(xin);
    eout = norm(A*(xin+noisein)-xout)./norm(xout);
    S(kk) = eout./ein;
end

figure(1);hist(S,50);
    %histogram of the stability of the forward problem
%
%Statistics for the 'inverse problem'
%----------------
%Inverse problem is : Solve xin = inv(A)*xout
%
invA = inv(A);
for kk = 1:numiter
    noisein = noiselevel*randn(dim,1);
    ein = norm(noisein)./norm(xout);
        %Note that the input and output are reversed...
    eout = norm(invA*(xout+noisein)-xin)./norm(xin);
    S(kk) = eout./ein;
end

figure(2);hist(S,50);
    %histogram of the stability of the inverse problem
xout_round = round(1e3*xout)./1e3;
xin_bad = invA*xout_round %Shows instability to rounding
```

Appendix B

Barrier Functions of Complex Variables

Consider the following barrier function $\phi(\xi) : \mathcal{C}^n \rightarrow \mathcal{R}$

$$\begin{aligned} \phi(\xi) &= -log(-f(\xi)) && \text{(B.1)} \\ &= -\log\left(-\frac{\xi^\dagger\eta + \eta^\dagger\xi}{2}\right) && \text{(B.2)} \end{aligned}$$

where $f : \mathcal{C}^n \rightarrow \mathcal{R}, \xi \in \mathcal{C}^n, \eta \in \mathcal{C}^n$. We use our $\mathcal{C}^n \rightarrow \mathcal{R}^{2n}$ transformation (eqn. (4.37)) and re-express in terms of the new variables.

$$[\xi] \rightarrow \begin{bmatrix} \Re(\xi) \\ \Im(\xi) \end{bmatrix} \equiv x \qquad \text{(B.3)}$$

$$[\eta] \rightarrow \begin{bmatrix} \Re(\eta) \\ \Im(\eta) \end{bmatrix} \equiv y \qquad \text{(B.4)}$$

where $(x, y) \in \mathcal{R}^{2n}$. Transforming to $\mathcal{R}^{2n}$ and using eqn. (4.60) and (4.61), we find

$$\begin{aligned}
\phi(x) &= -\log(-g(x)) = -\log(-x^T y) && \text{(B.5)}\\
\nabla g(x) &= y && \text{(B.6)}\\
\nabla \phi(x) &= \frac{1}{x^T y} y && \text{(B.7)}\\
\nabla^2 \phi(x) &= \frac{1}{(x^T y)^2} \nabla g(x) \nabla g(x)^T && \text{(B.8)}\\
&= \frac{1}{(x^T y)^2} y y^T && \text{(B.9)}
\end{aligned}$$

Let $(x^T y)^{-2} \equiv \alpha \in \mathcal{R}$, since it is just a scalar. The Hessian can be expressed as:

$$\begin{aligned}
\nabla^2 \phi &= \alpha y y^T && \text{(B.10)}\\
&= \alpha \begin{bmatrix} \eta_r \\ \eta_i \end{bmatrix} [\eta_r^T \eta_i^T] && \text{(B.11)}\\
&= \alpha \begin{bmatrix} \eta_r \eta_r^T & \eta_r \eta_i^T \\ \eta_i \eta_r^T & \eta_i \eta_i^T \end{bmatrix} && \text{(B.12)}\\
&\neq \alpha \begin{bmatrix} \Re(H) & -\Im(H) \\ \Im(H) & \Re(H) \end{bmatrix} && \text{(B.13)}
\end{aligned}$$

In the final step, we show that this Hessian matrix is not equivalent to any $H \in \mathcal{C}^{n \times n}$ based on the $\mathcal{C}^n \rightarrow \mathcal{R}^{2n}$ transformation rules for matrices (eqn. (4.47)). Therefore, regardless of how we define the generalized complex gradient or Hessians, we cannot include this form of barrier function in the complex domain using normal matrix manipulation rules. We are forced to perform the cumbersome transformation to real variables, despite suggestions found elsewhere [83, 25].

Appendix C

Fourier Transforms

In chapter 2, we examined the Helmholtz equation in the plane wave basis, and found that transforming into the Fourier domain played a central role. We also alluded to some problems associated with the truncation to finite bandwidth. In this chapter, we look more closely at Fourier transforms, and in particular numerical implementations of the Fourier transform as it applies to physics applications.

Fourier analysis is included in most undergraduate level curriculum, so it may seem strange to have it included in the body of the thesis. The use of Fourier transforms, both the continuous and discrete forms are ubiquitous in physics and engineering. They are used extensively in signal processing, image processing and compression, pattern recognition, and solutions of PDEs using spectral methods (as we use them here). Most computational software such as *Matlab*, *Mathematica*, and *Maple* all have built-in functions to compute these transforms. However, perhaps because it is so commonplace, there is greater risk that one does not first think more carefully about the underlying physics and will just let the machine grind through the calculation. The only warning I remember as an undergrad in learning about Fourier analysis was 'aliasing', and we were simply told to make sure we sample above the Nyquist frequency. However, that is not a viable option with photonic crystals, because the discontinuities imply an infinite Nyquist frequency. What exactly happens to this 'aliasing' behavior then? It turns out that there are other issues with numerical implementations of the Fourier transform, particularly for those who wish to use discrete Fourier transforms in computational physics. The goal of this chapter is

to bring awareness to some of these issues exacerbated by these discontinuities, and justify interpreting our results in the smooth function limit which we presented in Part II.

Organization

We will begin in section C.1 with some general bookkeeping by introducing the notation we will follow in this chapter for Fourier transforms, and then revisiting the Born von-Karman boundary conditions by paying closer attention to the topology of the direct space and Fourier space. In section C.2 we introduce formally the discrete Fourier transform, and in particular examine the fast Fourier transform (FFT) and some of its properties. There are some aspects of the FFT that are unnatural to physics applications (where the origin is usually at the center of the spectrum). This will lead to a discussion of symmetries in section C.3, where we see real discrepancies between what we naïvely think we are modeling, as opposed to what the mathematics is actually modeling. We also find a fundamental conflict in the symmetry of the physics and the symmetry as a result of the discretization. In light of the issues revealed in these sections, we emphasize the fundamental importance of considering the continuous function limit and explore the behavior for various transform schemes. As mentioned in section 2.4, we describe Li's insight into the convergence issues in the plane wave Helmholtz equation with Fourier factorization rules in section C.4, followed by concluding remarks in section C.5.

C.1 Boundary Conditions and Fourier Space

Consider a continuous function $f(\mathbf{r})$. We define the Fourier transform of the function as

$$F(\mathbf{k}) = \frac{1}{V}\int_V f(\mathbf{r})e^{-i\mathbf{k}\cdot\mathbf{r}}d\mathbf{r} \tag{C.1}$$

where V denotes the relevant integration volume. We can similarly define an inverse Fourier transform as

$$f(\mathbf{r}) = \int_{V_k} F(\mathbf{k}) e^{+i\mathbf{k}\cdot\mathbf{r}} d\mathbf{k} \tag{C.2}$$

There are many conventions for where to place the normalization factor $\frac{1}{V}$, and physicists tend to prefer the symmetrized form, but we have chosen to follow the convention [3] used in the treatment of the photonic bands problem. In the absence of any periodicity that extends to infinity, the Fourier transform of an arbitrary real-space function will be a continuous function in Fourier space. When we impose the Born von-Karman periodic boundary conditions, as we saw in section 2.2, we are left with a discrete set of allowed $\mathbf{k}$-vectors $\{\mathbf{k}\}$. More precisely, we have $F(\mathbf{k}') \equiv 0$ for any $\mathbf{k}' \notin \{\mathbf{k}\}$. We mentioned that imposing periodic boundary conditions is a standard approximation, and is considered valid in the tight-binding approximation. We explicitly examine the difference with the following example in 1D. Consider the rectangular function

$$f_1(x) = \begin{cases} A & \text{if } |x| \leq a, \\ 0 & \text{if } |x| > a. \end{cases} \tag{C.3}$$

The Fourier transform of $f_1(x)$ is:

$$\begin{aligned} F_1(k) &= \int_{-\infty}^{+\infty} f_1(x) e^{-ikx} dx & \text{(C.4)} \\ &= 2A \frac{\sin(ak)}{k} & \text{(C.5)} \end{aligned}$$

where we have ignored the normalization in eqn. (C.1) to facilitate comparison with the periodic case.

If we now introduce an artificial periodicity by defining a supercell of length $10 \times 2a$,

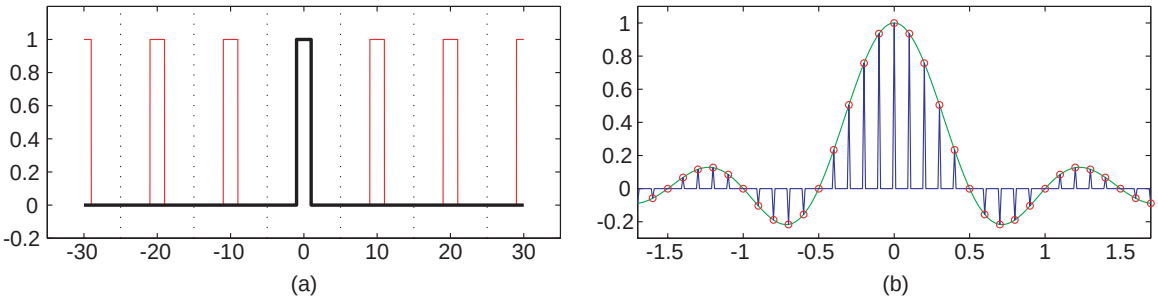

Figure C.1: In (a) we show $f_1(x)$ in black, and $f_2(x)$ with the artificial periodic boundary conditions in red. The dashed line helps visualize the periodicity of f_2. In (b), we plot the Fourier transform of f_1 and f_2. F_1 is plotted in green and the allowed k values of F_2 are shown as red circles. We plot in the blue curve the underlying continuous form of F_2 in k space. The 'delta function train' as a result of the real-space periodicity is shown explicitly this way.

we can define

$$f_2(x) = \begin{cases} A & \text{if } |x| \leq a, \\ 0 & \text{if } a < |x| < 5a. \end{cases} \tag{C.6}$$

$$f_2(x + Nx_p) = f_2(x) \tag{C.7}$$

where N is an integer, and $x_p \equiv 10a$ is the periodicity introduced. The Fourier transform of this function has the same form as $F_1(k)$ with the exception that it is modulated with a delta function 'comb' corresponding to the allowed k values. In figure C.1, we plot the two functions in real space and Fourier space.

The ratio of the two normalization factors we omitted accounts for the extra features in $f_2(x)$ that are absent in the original function $f_1(x)$. However, for applications such as spectral analysis, this is unimportant since it is the relative magnitude of the different frequency components that matter. Quantities involving continuous k-space integrals are well approximated by Riemann sums. We see in figure C.1 that we do not lose information by incorporating the artificial boundary condition. Particularly for functions like $f_1(x)$ that are negligible outside of the supercell (in our case the function is identically zero), we see that this is a very good approximation. One of

our Fourier transform pair becomes a discrete sum rather than an integral:

$$F(\mathbf{k}) = \frac{1}{V_{sc}} \int_{V_{sc}} f(\mathbf{r}) e^{-i\mathbf{k}\cdot\mathbf{r}} d\mathbf{r} \tag{C.8}$$

$$f(\mathbf{r}) = \sum_{\mathbf{k}} F(\mathbf{k}) e^{+i\mathbf{k}\cdot\mathbf{r}} \Delta\mathbf{k} \tag{C.9}$$

where V_{sc} indicates the volume of the supercell.

C.2 Numerical Implementation

With arbitrary functions, we often cannot perform the integration analytically. Numerical implementation of the Fourier transform takes the form of a discrete Fourier transform (DFT), and one of the most common algorithms for its implementation is the fast Fourier transform (FFT). In a FFT, the real space function is sampled at regular intervals, and the number of plane waves is truncated to allow numerical evaluation of the transform. The number of points used in the real-space sampling N matches the number of plane waves.

With the identification from $f(x)$ to f_x, the FFT relations are defined to be:

$$\begin{aligned} F_k &= \sum_{x=0}^{N-1} f_x \exp\left(\frac{-2\pi i}{N} kx\right), \text{ Forward FFT} \\ f_x &= \frac{1}{N} \sum_{k=0}^{N-1} F_k \exp\left(\frac{2\pi i}{N} kx\right), \text{ Inverse FFT} \end{aligned} \tag{C.10}$$

where k and x are now integer values from 0 to $N-1$ in units of their respective basis vectors. In general 3D form, let $\{\hat{s_1}, \hat{s_2}, \hat{s_3}\}$ define the periodicity of the lattice. The unit vectors in real-space becomes $\{\hat{x_1}, \hat{x_2}, \hat{x_3}\} \equiv \{\hat{s_1}/N_1, \hat{s_2}/N_2, \hat{s_3}/N_3\}$ where N_1, N_2, N_3 are the number of sampling points along each direction. The basis vectors

in **k**-space become:

$$\mathbf{b}_1 = 2\pi \frac{\mathbf{s}_2 \times \mathbf{s}_3}{\mathbf{s}_1 \cdot (\mathbf{s}_2 \times \mathbf{s}_3)} \tag{C.11}$$

$$\mathbf{b}_2 = 2\pi \frac{\mathbf{s}_3 \times \mathbf{s}_1}{\mathbf{s}_2 \cdot (\mathbf{s}_3 \times \mathbf{s}_1)} \tag{C.12}$$

$$\mathbf{b}_3 = 2\pi \frac{\mathbf{s}_1 \times \mathbf{s}_2}{\mathbf{s}_3 \cdot (\mathbf{s}_1 \times \mathbf{s}_2)} \tag{C.13}$$

Observe as the real-space periodicity grows ($|s_i| \to \infty$), the resolution increases in **k**-space (i.e. $|b_i| \to 0$). The high frequency cutoff is precisely at Nyquist when we use the same number of points in both **k**-space and real-space.

One reason for its popularity is that the computational complexity for the FFT scales as $O(N \log N)$ due to the Cooley-Tukey algorithm [84], compared to $O(N^2)$ for a direct evaluation of the sum. In creating the MPB package, Steven Johnson et al. have also put out a free package [85] for the FFT called FFTW (Fastest Fourier Transform in the West).

There are some notable properties about the FFT we should highlight. First, the normalization factor is absorbed in the inverse transform, which is different than the convention we have adopted in the continuous case. This unfortunate discrepancy means we have to be more careful with our bookkeeping of the normalization factors, but otherwise poses no problems. The other property which usually affects physicists is that the FFT convention defines the domain of x and k such that the 0 value is the first element rather than at the center of the spectrum (see eqn. (C.10)).

In physics however, the origin is usually defined at the center of most problems that we analyze to more easily exploit symmetries. Computing the FFT with any standard software package using discretized data in 'physics order' will give incorrect results. In figure C.2 we show graphically what is effectively the conventional FFT ordering. The correct function $f(x)$ one needs to use in order to get what physicists think they should get has the 'negative' half of the data translated by the periodicity imposed. This translation of the spectrum in both k-space and real space is simply a substitution of $x \to x + N$ and $k \to k + N$ respectively. In eqn. (C.10), this term

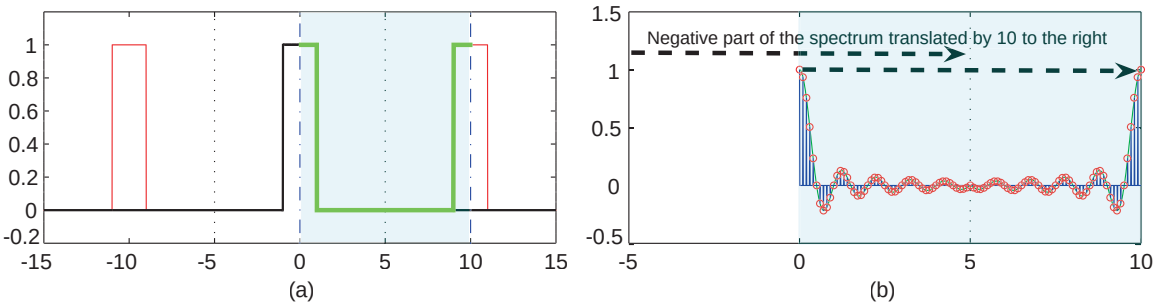

Figure C.2: In (a) we again show $f_1(x)$ in black, and $f_2(x)$ in red. The blue dash-dotted line defines the new domain (shaded in turquoise) of x under the FFT convention. To use a standard FFT routine, we need to use the green curve rather than the black curve to get the right results. On the right, (b) shows the equivalent k-space domain for the Fourier transform. The $k < 0$ part of the spectrum is translated over as shown. The turquoise region shows k-space domain under the FFT convention.

appears in the exponential and is evaluated to 1, so we see that the resulting FFT is left unchanged by this translation. To be completely transparent, this means that for a given discretized f_x, if we attempt to compute the FFT coefficient F_{k+N}, we get identically F_k. The generalization to 2D is illustrated graphically in figure C.3.

By defining the computational grid in FFT order rather than in physics order, we can take advantage of the FFT algorithm without shifting the indices around. In matlab, there are built-in functions that do this called *fftshift*, which go from FFT order to physics order, and *ifftshift*, which go from physics order to FFT order. If the FFT is used extensively, it is more convenient to simply define the grid in the FFT order, and shift only when plotting the results.

This shift property is reminiscent of the concept of the reduced zone scheme of **k**-space representation in solid state physics (see Chapter 9 in [12]) for periodic potentials. However, we will see in the following section that the FFT symmetry is artificial and one needs to be very careful how one treats these terms in the context of computational physics.

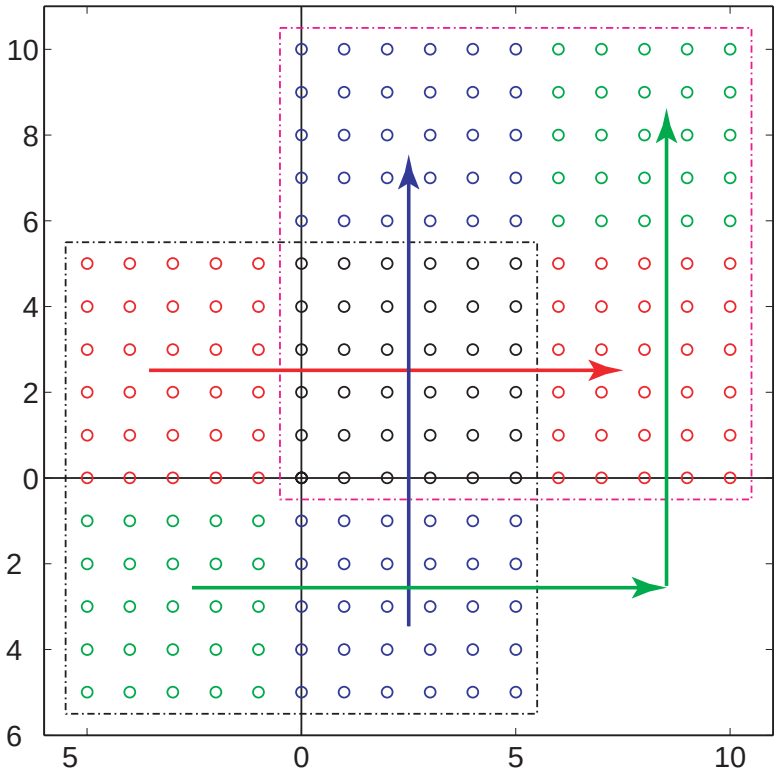

Figure C.3: The four quadrants of a 2D FFT are shifted from physics ordering to FFT ordering as shown. The black dash-dotted line indicates the usual physics domain, while the magenta dash-dotted line indicates the nominal FFT domain.

C.3 The Symmetry Problem

Suppose we define a 1D computational grid with $N = 2n + 1$ points such that $\{k\} = \{-n, -n+1, -n+2, \ldots, n-2, n-1, n\}$, and position $\{x\} = \{-n, -n+1, -n+2, \ldots, n-2, n-1, n\}$. For simplicity, consider the Helmholtz operator (eqn. (2.28)) in 1D, and notice the terms of the form $\eta_{k-k'}$. This has the form of a Toeplitz matrix (i.e. matrices of the form $A_{m,n} = A_{m-n}$), which has a natural connection mathematically with the FFT [86]. The Toeplitz matrix has elements like η_κ, where $n < |\kappa| \leq 2n$, exceeding our defined k-point domain. For simple geometries, one could use the expression for the analytical Fourier transform. For more complicated geometries, we might be tempted to use the FFT symmetry and equate $\eta_\kappa = \eta_{\kappa \pm n}$ (depending on the sign of κ), since, as we saw earlier, they are formally equivalent mathematically. This would give us the circulant form of the Toeplitz matrix. How-

ever, we presumably have truncated the grid at the chosen size because the omitted high frequency components are 'small enough', whereas these η_κ terms can be quite large. So how do we resolve the discrepancy on how to handle these Fourier coefficients that lie outside our computational domain?

C.3.1 The underlying real-space function

The key is in recognizing that the real-space function is fundamental because it is the one that corresponds to a physical quantity. With regards to the FFT symmetry, recall that the valid but artificial periodicity in the real-space function gave rise to the delta-function lattice in **k**-space. We justified the approximation by invoking the tight-binding approximation. If we now insist on an artificial periodicity in **k**-space, then necessarily (by the symmetric nature of the Fourier transform) we enforce a convolution of any real-space function with a delta-function lattice as well. This means that our model of the continuous function is identically zero everywhere except the points we happen to be sampling (see figure C.4a).

This illustrates that we must not use the FFT symmetry to determine the correct coefficients for terms outside of our computational domain. We could (as we might with the analytical Fourier coefficients) do an oversampled transform with $N \times 2^{n_D}$, where n_D is the number of dimensions in real-space to evaluate those coefficients. The problem with this approach is that it is no longer self-consistent with our truncation condition. When we chose the computational domain, for better or for worse, we effectively set *a priori* any Fourier component exceeding $k_{\max}$ equal to zero. Since other quantities have the same bandwidth limitation, self-consistent treatment implies setting those ϵ_k equal to zero as well.

Given the analysis here, it is now obvious that the underlying real-space function for some set of FFT coefficients can be quite different than what we think they represent. Given our strict truncation approximation, we now examine the 1D rectangular function again. This time, we take an N point FFT of the rectangular function, then transform back to real-space with a $2N$ point FFT by explicitly zero padding the

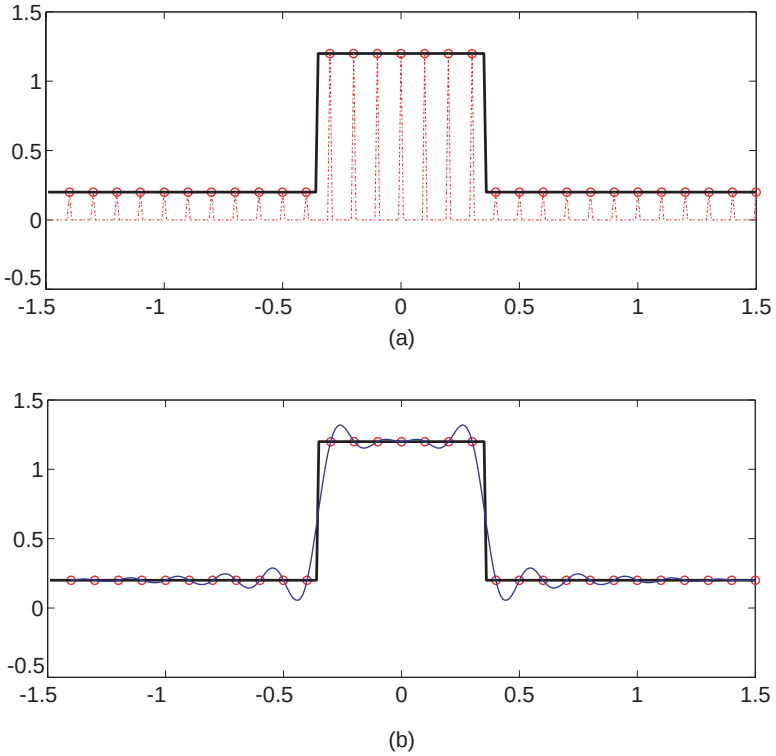

Figure C.4: (a) The underlying continuous function of a nominal rectangular function allowing the FFT symmetry. (b) The underlying continuous function of a nominal rectangular function reconstructed from the N FFT coefficients with hard truncation.

k-components outside the original bandwidth. Figure C.4b shows the true representation of an FFT. Even though the FFT/IFFT pair seems to perfectly reconstruct a discontinuous jump, an examination of the underlying continuous function shows the misrepresentative sampling that actually happens. Note also that after the hard truncation we no longer have a Toeplitz matrix because of our FFT preferred ordering, which is different from the other works that use the PWE method and keep $\eta_{\mathbf{k}-\mathbf{k}'}$ (or the equivalent $\frac{1}{\epsilon}$ term) as a Toeplitz matrix[11, 14, 87, 88, 20].

C.3.2 Even vs. odd

A final problem with the FFT symmetry that is quite subtle shows up in how we choose to discretize the grid in $\mathbf{k}$-space. As before, the spacing is strictly determined by the real-space periodic BCs, while $\mathbf{k}_{\max}$ is chosen according to some truncation

condition. The final detail addresses the boundary of the **k**-space grid, meaning a determination of whether an even or an odd number of **k**-points are used along each **k** direction. It is known that the FFT algorithm is most efficient when N is a power of 2 or a product of small prime numbers [85]. Usual implementations choose N to be a power of 2, which is clearly even. Here, I argue that an odd-numbered grid is the correct choice for self-consistency considerations, particularly when we have high frequency components we have had to truncate out.

Consider now in 1D an N_o point transform where $N_o = 2n+1$ and an N_e transform where $N_e = 2n$ of the same periodic continuous function in real-space. Shifting back into a physics preferred coordinate system, our k-space will have in the odd case $\{k_o\} = \{-n, -n+1, \ldots, n-1, n\}$. In the even case, we have a choice of either $\{k_e\} = \{-n, -n+1, \ldots, n-2, n-1\}$ or $\{k_e\} = \{-n+1, -n+2, \ldots, n-1, n\}$, and the two are equivalent because of the FFT symmetry, i.e. any $F_{-n} = F_{N-n}$. Consider further a real-valued real-space function (such as a dielectric function). The continuous Fourier transform symmetry in **k**-space is:

$$F_{\mathbf{k}} = F^*_{-\mathbf{k}} \tag{C.14}$$

This illustrates that the physical symmetry conflicts with the FFT symmetry which arises as a result of discretizing. We do not have this conflict when we choose an odd numbered FFT, since F_n and F_{-n} are independent. With an even-numbered FFT of a real-valued function, this constrains the boundary **k** elements to take on only real values. In 1D, this may not be significant, since there is only one element. In 2D, the number of boundary elements increase, and we show in figure C.5 the conflict in symmetry.

For well-behaved functions where our sampling rate is well above Nyquist, this conflict is not significant, simply because F_n is near zero. If we do not sample at a sufficient rate (e.g. when we have discontinuities), the conflict becomes much more significant. In figure C.6 we show an arbitrary discontinuous function in 2D that can be a possible PBG dielectric function. We take the 2D FFT using an $N \times N$

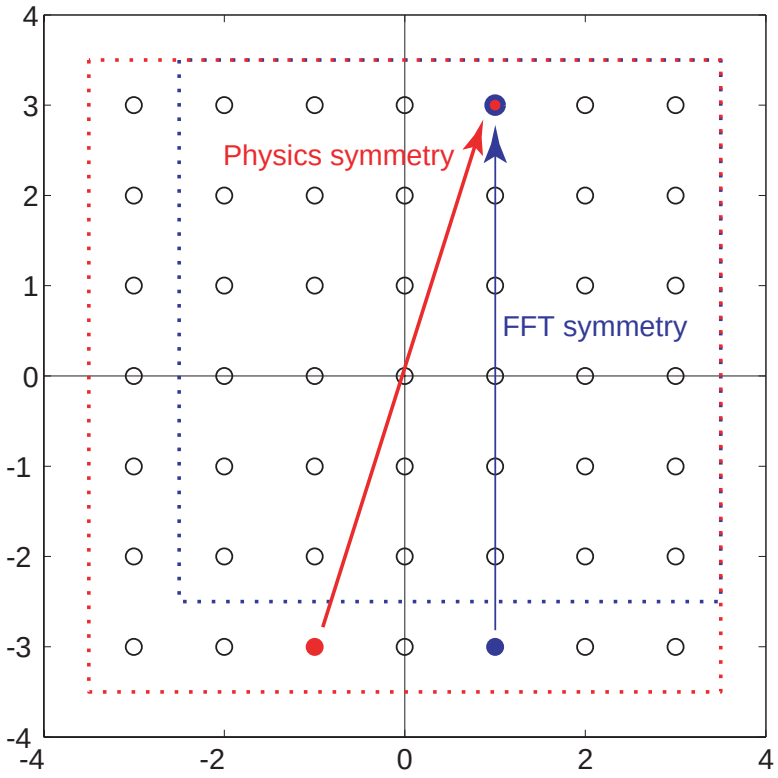

Figure C.5: 2D **k**-space diagram showing the effect of the FFT symmetry on Fourier coefficients at the boundaries

grid for N even and N odd. We show the resulting spectra of coefficients in figure C.7. Our discretization choice affects not only the boundary values, but as shown in the plot, there is a significant discrepancy between the two schemes in even the largest of the coefficients (i.e. where the **k**-vector is close to the origin, far away from the truncation limit). Since the odd numbered grid preserves the proper number of degrees of freedom and symmetries, we consider it the discretization scheme that is actually more appropriate for computational physics, especially when modeling discontinuities.

C.4 Fourier Factorization

A final remark we will make about Fourier transforms as it applies to the PBG problem deals with the Fourier coefficients of a product of two functions. Consider a

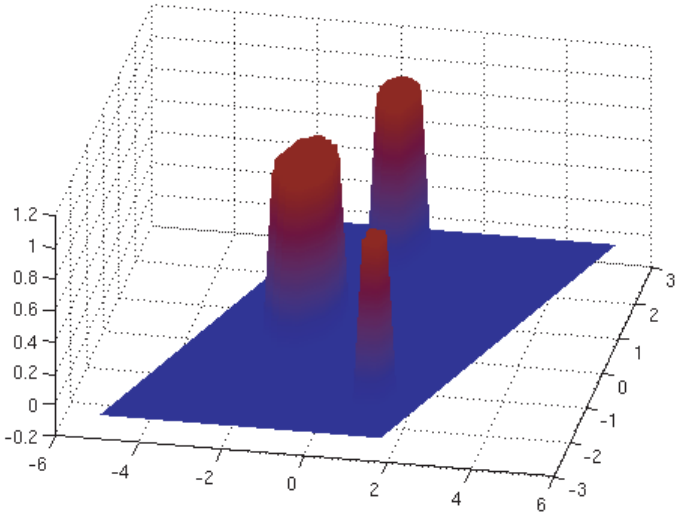

Figure C.6: A 2D real-space function with discontinuities that is representative of an arbitrary PBG dielectric function.

function $f(x) = g(x) \cdot h(x)$. The problem is to find the Fourier coefficients F_k, given G_k and H_k, the Fourier coefficients of $g(x)$ and $h(x)$ respectively. This can be done using *Laurent's Rule* such that:

$$F_n = \sum_{m=k_{\min}}^{k_{\max}} G_{n-m} H_m \tag{C.15}$$

where G_{n-m} is the familiar Toeplitz matrix. However, suppose $g(x)$ and $h(x)$ are functions with *concurrent* discontinuities at x_d, and suppose further that the discontinuities at x_d are *complementary* such that $f(x)$ is continuous at x_d, i.e.

$$\lim_{x \to x_d^+} f(x) = \lim_{x \to x_d^-} f(x) = f(x_d) \tag{C.16}$$

We encounter this situation with our source free non-magnetic geometry. The magnetic fields are continuous everywhere, which implies (by eqn. (2.3)) that $\mathbf{D}$ is continuous. Since $\epsilon(\mathbf{r})$ has discontinuities, $\mathbf{E}(\mathbf{r})$ must have complementary and concurrent discontinuities such that their product is continuous.

Li proved [18] that even as we take the set $\{\mathbf{k}\}$ to infinity, Laurent's rule will never

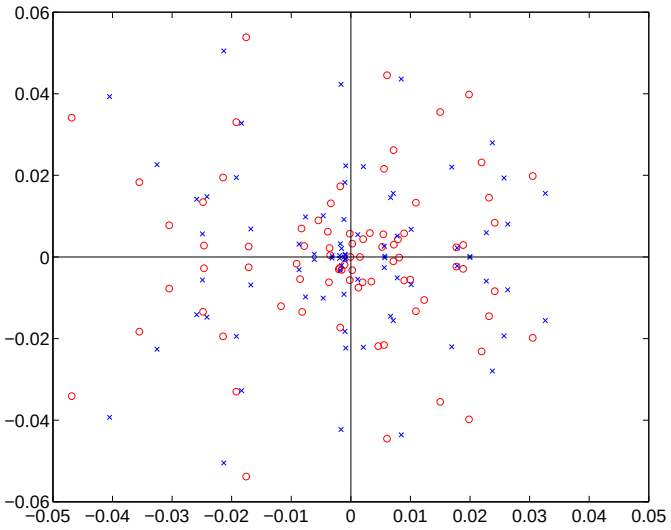

Figure C.7: The value of the Fourier coefficients for an even numbered FFT vs. an odd numbered FFT are plotted on the complex plane, i.e. the axes are the real and imaginary parts of F_k of the function shown in figure C.6. Notice the discrepancy in most of the points. We have plotted the 81 **k**-points with the largest magnitude. Note that the bandwidth for both geometries are the except for the boundary point, showing the two schemes are not self-consistent.

converge. Instead, he applied what is now known as the *Inverse Rule*:

$$F_n = \sum_{m=k_{\min}}^{k_{\max}} \left[\Gamma_{n-m}\right]^{-1} H_m \tag{C.17}$$

where Γ_k is the Fourier expansion of $\gamma(x) = \frac{1}{g(x)}$. The superscript $^{-1}$ denotes matrix inversion. The difference in convergence is shown in figure C.8.

Given Li's analysis, it is no longer surprising why the convergence of the PWE method depends on the polarization of the field and how we treat the dielectric Toeplitz matrix. Rigorous application of his factorization rules to 2D and 3D has spawned the fast Fourier factorization methods (cf. [87, 20]), but the physics is ultimately embodied by the effective medium approach [17, 4].

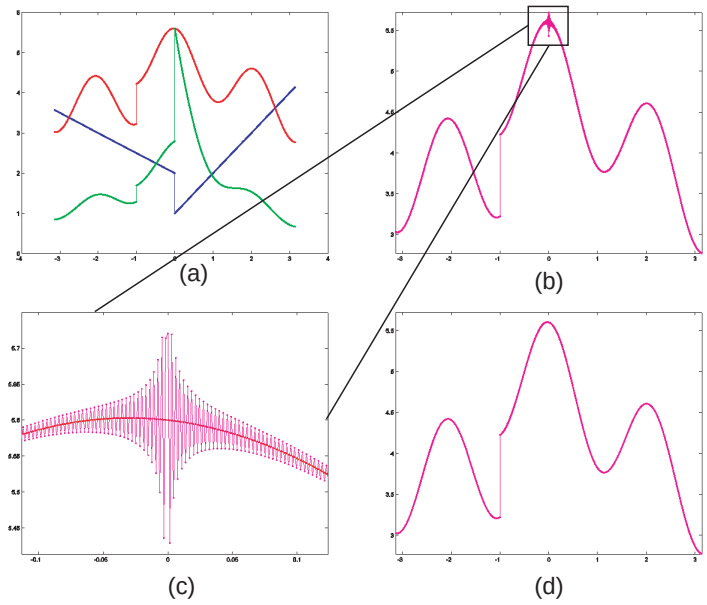

Figure C.8: Illustration of Laurent's rule and the inverse rule. (a) Given the Fourier coefficients of two functions $g(x)$ in blue and $h(x)$ in green, we want to find the product $f(x)$ in red. Notice a single concurrent and complementary discontinuity at $x = 0$, and a concurrent but non-complimentary discontinuity at $x = -1$. (b)Result using Laurent's rule. Notice the non-concurrent discontinuity had no trouble converging. (c) Magnification of problem area (d)Good convergence of result using Li's inverse rule.

C.5 Conclusion

In this appendix, we explored some of the intricacies of the numerical implementation of Fourier transforms. For functions with discontinuities, our inherent inability to satisfy the Nyquist criterion exacerbates the issues discussed in this chapter. For these reasons, we cautioned against blind acceptance of the FFT/IFFT output, but instead return to a continuous function limit interpretation. Given the accuracy limitations revealed in chapter 2 and this appendix, self-consistency in modeling should be the primary focus. We therefore give up on the notion of accurately modeling dielectric slabs with etched air holes. In chapters 5–8 where we describe geometries of structures, we use language such as a *nominal* geometry consisting of a lattice of air holes of radius $0.3a$ and so forth. It is to be understood that we are really talking about the real-space continuous function representation of the truncated Fourier series.