CHAPTER 4. IMAGE-PROCESSING TECHNIQUES

In Chapter 3, we saw that full-wave numerical techniques can accurately simulate radiating structures in fine detail. The model we created for a microstrip patch antenna included the effects of a finite ground plane and substrate as well as a feed structure. The major difficulty that was posed was how to understand the large data set that was generated by the computational code. By employing visualization techniques, we were able to view the data from a global perspective. These techniques allowed us to interpret the data set efficiently. However, there may be cases where simply viewing a data set may not suffice. For instance, we may need to analyze an antenna that supports several different propagating waves (or modes). By using the techniques presented in Chapter 3, a superposition of the various modes would be displayed. This can be quite confusing, especially when one mode is much larger in amplitude than the other modes. Due to dynamic range problems, the effect of the lower amplitude modes may be masked when the information is displayed concurrently. If we could devise a tool that would allow us to analyze the effect of each mode independently, the task of interpreting the data could be made easier.

To develop such an analysis tool, we must first recognize that each frame of visualized data is essentially an image. If we further realize that each mode has a unique propagation constant, then we can use image-processing concepts to distinguish and filter the various modes based on their propagation characteristics. In the following sections we will detail how this can be done. We start by reviewing discrete Fourier transform principles and Gabor filtering in Sections 4.1 and 4.2, respectively. Then, in Section 4.3, we explain how these concepts are applied to our particular problem. Finally, in Section 4.4 we present several examples of strategies for determining wave properties.

4.1 Discrete Fourier Techniques

Both continuous and discrete Fourier transforms are well-established analysis techniques. These methods are frequently utilized in the disciplines of applied mathematics and engineering. By employing Fourier techniques, we can understand the time and frequency characteristics of a mathematical function x(t). If the values of x(t) are defined over the entire real line ℝ, then the conversion between the time and frequency domain is accomplished using the integral form of the Fourier transform equations [16]:

$$\begin{aligned} \hat{X}(f) &= \int_{-\infty}^{\infty} x(t)e^{-j2pft}dt \\ x(t) &= \int_{-\infty}^{\infty} \hat{X}(f)e^{+j2pft}df \end{aligned} \quad . \tag{4-1}$$

However, in many applications the function x(t) is only known at discrete points in time. This is the case when we deal with data generated by a computational algorithm or measurement process. If the function x(t) is sampled at evenly spaced intervals, then the integral form of (4-1) reduces to a summation and we obtain the following discrete Fourier transform (DFT) equations [16]:

$$\begin{aligned} \hat{X}_k &= \sum_{n=0}^{N-1} x_n e^{-j\frac{2pkn}{N}} \\ x_n &= \frac{1}{N}\sum_{k=0}^{N-1} \hat{X}_k e^{+j\frac{2pkn}{N}} \end{aligned} \quad . \tag{4-2}$$

We should note that in most applications, the fast-Fourier transform (FFT) algorithm is used to calculate the discrete transform instead of the DFT equations given above. The properties and results of these two methods are identical, although the FFT is computationally more efficient. Unfortunately, the FFT algorithm is fairly complicated and cannot be written in a compact form. Therefore, we list the equations for the DFT and refer the interested reader to [17] and [18] for details of the FFT.

Now that we have defined the Fourier transform equations, we are ready to discuss their properties. The FDTD code that we use throughout this thesis provides discrete-time data. Therefore, we will focus on the properties of the FFT, which are listed in Table 4-1. There are a number of sources that provide proofs of the properties that are listed for the FFT. So, we will simply report the results here and refer the interested reader to [17] and [18] for the derivations.

In many cases, the properties of the FFT can be exploited to improve the computational efficiency of an algorithm. For instance, by using the symmetry properties given in Table 4-1 we can calculate the FFT of two real-valued functions with a single FFT. To do so, we place one function (length N) in the real part and the other function (also length N) in the imaginary part of a data array and evaluate the FFT. We can then separate the two transforms using properties 4 and 5 of the table. Full details of this procedure, along with many others, are provided in [19].

In addition to the above properties, the FFT also exhibits an assumed periodicity in the time domain. In other words, the windowed data is assumed to continuously repeat with a period equal to the window length (N). In the time domain this property can result

Finite-Length Sequence (length N)	DFT
1. x_n	$\hat{X}_k$
2. y_n	$\hat{Y}_k$
3. $ax_n + by_n$	$a\hat{X}_k + b\hat{Y}_k$
4. x_n real	$\hat{X}_{N-k} = \hat{X}_k^{\ *}$
5. x_n imaginary	$\hat{X}_{N-k} = -\hat{X}_k^{\ *}$
6. $x_{N-n} = x_n^{\ *}$	$\hat{X}_k$ real
7. $x_{N-n} = -x_n^{\ *}$	$\hat{X}_k$ imaginary
8. $x_n^{\ *}$	$\hat{X}_{-k}^{\ *}$
9. $x_{-n}^{\ *}$	$\hat{X}_k^{\ *}$
10. x_{n+m}	$\hat{X}_k e^{j\frac{2pkm}{N}}$
11. $x_n e^{-j\frac{2p\,\ln}{N}}$	$\hat{X}_{k+l}$
12. $x_n y_n$	$\frac{1}{N}\sum_{l=0}^{N-1}\hat{X}_l\hat{Y}_{k-l}$
13. $\sum_{m=0}^{N-1} x_m y_{m+n}$	$\hat{X}_k\hat{Y}_k^{\ *}$

Table 4-1. Important properties of the FFT [16] and [17].

in leakage, as we will illustrate with the example shown in Figures 4-1 and 4-2. In Figure 4-1 (a), we sample a 1 Hz cosine wave so that exactly 2 cycles appear in the data window. Graphing the magnitude results of the FFT in Figure 4-1 (b), we see that the frequency content of the signal is exactly as we would expect (peaks at ±1 Hz).

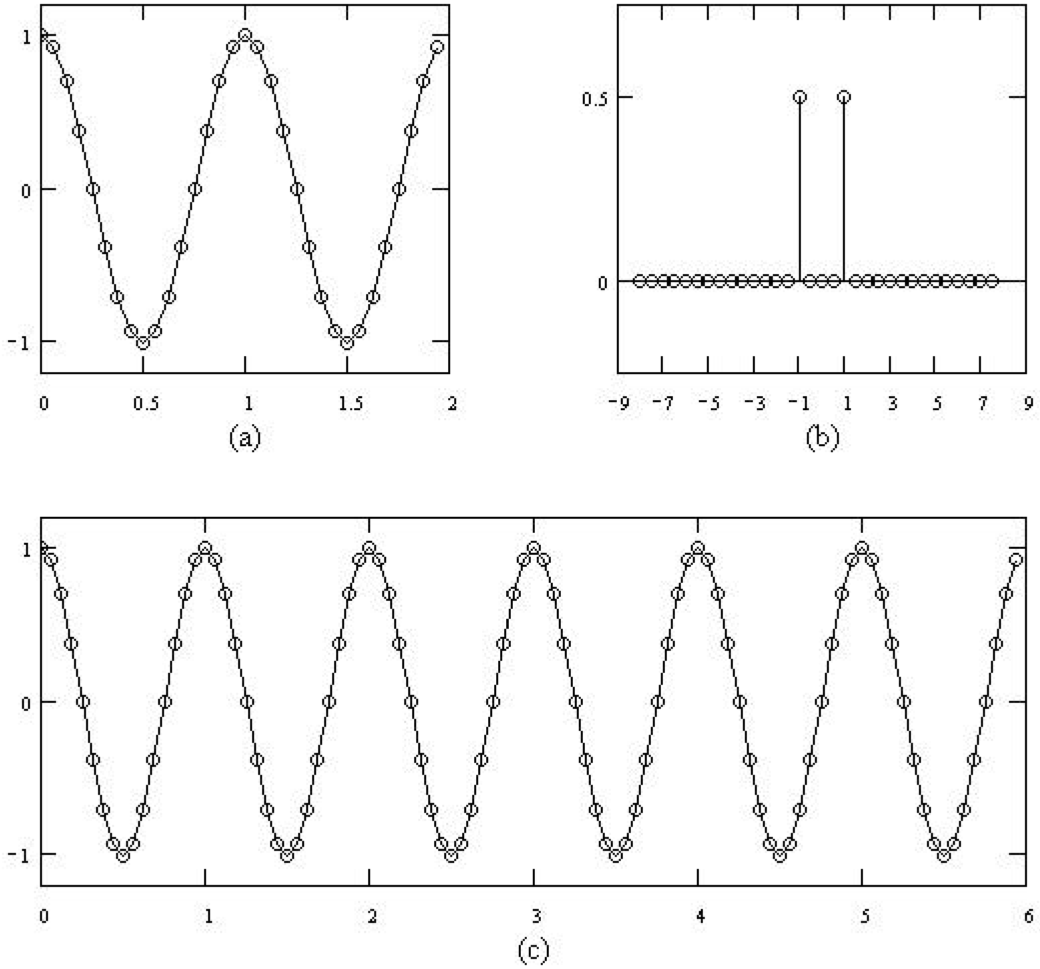


Figure 4-1. (a) Time domain data, (b) frequency-domain data, and (c) repeated windows for 2 cycles of a 1 Hz cosine wave sampled with Δt=0.0625 (N=32).

Next, in Figure 4-2 we sample a second cosine wave (1.25 Hz) so that exactly 2.5 cycles appear in the data window. Now when we calculate the FFT and plot the magnitude of the result, we would expect to see clearly defined peaks at ±1.25 Hz. Due to leakage, we obtain a result that is significantly different. To understand how to minimize the effects of leakage, we look at the periodically repeated windows shown in Figures 4-1,2 (c).

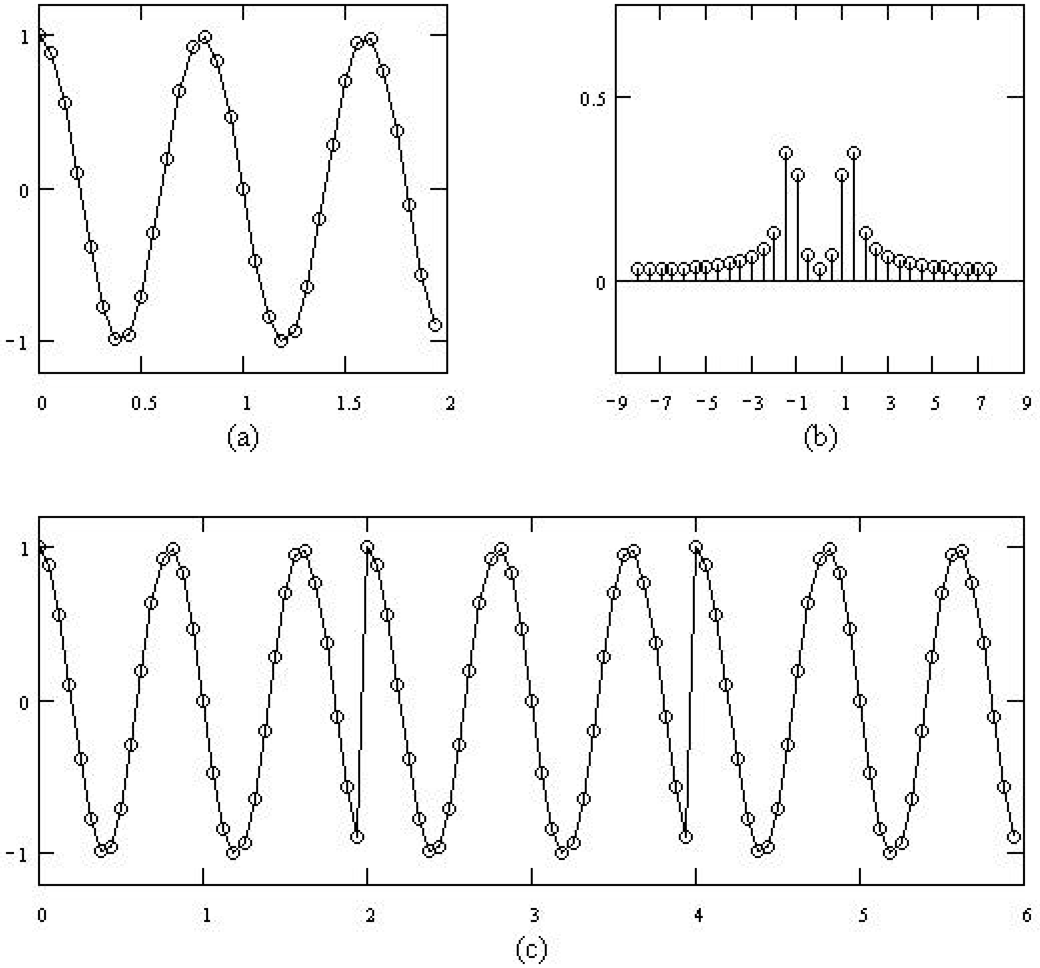


Figure 4-2. (a) Time domain data, (b) frequency-domain data, and (c) repeated windows for 2.5 cycles of a 1.25 Hz cosine wave sampled with Δt=0.0625 (N=32).

In the first case, the data window contains an integer number of cycles. Therefore, the last data value of each repeated window coincides with the beginning value of the next window. The second cosine wave contains a non-integer number of cycles. So when the windowed data is repeated, there is a recurrent discontinuity that is introduced at the edge of each window. According to Ramirez [16], we can reduce the effect of leakage by eliminating or reducing these discontinuities. In order to do so, we can merely shorten the window so that it contains only an integer number of cycles. This method works well for waveforms that are clearly periodic; however, many waveforms are not periodic. In these cases, we can use windowing techniques to taper the waveform at the edges of the window. This is a well-established technique in signal processing and there are a number of window functions that are commonly used, including Hamming, Hanning, Blackman, etc. The characteristics and parameters of these windows can be found in most signal processing handbooks, so we will simply list several references here [16] –[19].

The final issue that we need to address is that of the sampling theorem. Suppose we wish to sample a function x(t) that is bandwidth limited to frequencies smaller than the cutoff frequency f_c. According to sampling theory, we must discretely sample the function using a time interval that is given by

$$\Delta t \leq \frac{1}{2 f_c}. \tag{4-3}$$

If we allow Δt to exceed this critical Nyquist rate, then the spectral energy that lies outside the range ($-f_c$, f_c) will be spuriously moved into that region [19]. This phenomenon is commonly referred to as aliasing. Unfortunately, there is very little that can be done to correct aliasing once the signal is sampled. In fact, the only way to overcome aliasing is through the proper selection of Δt. If we know the frequency content of the function x(t), this task is trivial. However, in most cases we do not know this information. In these cases, we can pass the data through an analog filter that is of a known frequency response, which enforces a known limit on the data.

In this section we have defined a means of transforming discretely sampled data back and forth between the time and frequency domains. We are now prepared to develop a filter that will enable us to distinguish between various propagating modes. Luckily, we can borrow from image-processing techniques that are used to measure image velocities. The next section provides a full explanation.

4.2 Gabor Filters

As we saw in the previous section, Fourier transforms allow us to analyze the frequency characteristics of many mathematical functions. However, there are limitations as to the types of functions that the Fourier transform can handle. For instance, suppose we wish to analyze the frequency content of a piece of music. Traditional Fourier methods would lump all of the different frequencies over the entire piece together. While this does present us with some information about the song, it does not explain the timing of how the notes are played. In order to approach this problem we need to analyze the function locally. Towards this end, Dennis Gabor [20] developed a special window function that localizes signals in both time and frequency. We can see below that the Gabor filter is essentially a Gaussian window:

$$g(t) = \frac{1}{\sqrt{2p}\, s} e^{\left(\frac{-t^2}{2s^2}\right)} e^{jw_0 t}\ . \qquad (4\text{-}4)$$

Transforming the filter to the frequency domain, we obtain another Gaussian window that is centered at ω0

$$\hat{G}(w) = e^{\left[\frac{-(w-w_0)^2}{2\hat{s}^2}\right]}. \qquad (4\text{-}5)$$

From the above two equations, we can see the time and frequency localization of the Gabor filter.

Taking the time-frequency localization concept a step further, Gabor [20] used the Heisenberg uncertainty principle to calculate a generalized theoretical lower bound on the product of the temporal extent of a signal and the extent of its power spectrum. The exact lower bound depends on how the signal extents are defined in the time and frequency domains. If we define the window widths based on one standard deviation of the power distribution of the signal, then we obtain the result

$$\Delta_t \Delta_w \geq \frac{1}{2} \qquad (4\text{-}6)$$

where

$$\begin{aligned} \Delta_t &= \left[\frac{\int_{-\infty}^{\infty} t^2 \left|f(t)\right|^2 dt}{\int_{-\infty}^{\infty} \left|f(t)\right|^2 dt}\right]^{\frac{1}{2}} \\ \Delta_w &= \left[\frac{\int_{-\infty}^{\infty} w^2 \left|\hat{F}(w)\right|^2 dw}{\int_{-\infty}^{\infty} \left|\hat{F}(w)\right|^2 dw}\right]^{\frac{1}{2}} \end{aligned}\ . \qquad (4\text{-}7)$$

Plugging in the functional form of the Gabor filter, we find that the window provides optimal time-frequency localization. In other words, for a fixed window width in the time domain, the Gabor filter provides the finest frequency resolution. This is a primary reason why Gabor windows are desirable as filters.

Another nice feature about Gabor filters is the ease with which they can be extended to multiple dimensions. Using matrix notation, we can write the functional form of an n-dimensional Gabor filter. We begin by defining a signal f(X), where X is defined over $\mathbb{R}^n$, and its corresponding Fourier transform $\hat{F}(\Omega)$. Additionally, C and $\hat{C}$ represent the covariance matrices that correspond to $\left|f(X)\right|^2$ and $\left|\hat{F}(\Omega)\right|^2$. Now the Gabor filter and its transform can be written as [21]

$$\begin{aligned} g(X;C) &= \frac{1}{(2p)^{n/2}|C|^{1/2}} e^{\left(-\frac{1}{2}X^T C^{-1} X\right)} \\ \hat{G}(\Omega;\hat{C}) &= e^{\left(-\frac{1}{2}\Omega^T \hat{C}^{-1} \Omega\right)} \end{aligned} . \qquad (4\text{-}8)$$

Finally, the uncertainty relation is generalized to [21]

$$|C|^{1/2}\left|\hat{C}\right|^{1/2} \geq \frac{1}{2^n} . \qquad (4\text{-}9)$$

Gabor filters were originally applied in the area of communications. Gabor [20] was seeking to find an optimal utilization of frequency bands for communications channels. Recently, the Gabor filter has found a number of uses in the area of image-processing, including compression, enhancement, motion sensing, and many more [23]. One application of particular interest is the measurement of image velocity [21] and [22]. This technique converts image data to the frequency domain and then uses a Gabor filter to detect motion and calculate the corresponding velocity. We can apply a similar technique to our problem in electromagnetics. The various modes supported by an electromagnetic structure will each have its own characteristic propagation constant. By converting to the time domain, we should be able to distinguish and filter the various modes using a Gabor filter. The details of the overall technique are presented in the next section.

4.3 Topological Understanding of the Process

The above sections provide a review of discrete Fourier transform principles and introduce a filtering window that permits time-frequency localization. The purpose of this section is to explain how these two pieces fit together. We will provide specific details of the image-processing tools that will distinguish and filter the various propagating modes of an electromagnetic structure. Figure 4-3 illustrates how these tools will fit in with the visualization components.

As we can see, the image-processing tools operate on the raw data set straight from the FDTD computational code. We could have elected to work with the actual image data. However, if we recall, the post-processor transforms the raw data from the phasor-domain to the steady-state time-domain. This essentially adds another dimension to the data set and requires that we compute a three-dimensional rather than a two-dimensional FFT. Therefore, using the raw data set greatly reduces the computational requirements of the analysis tools.

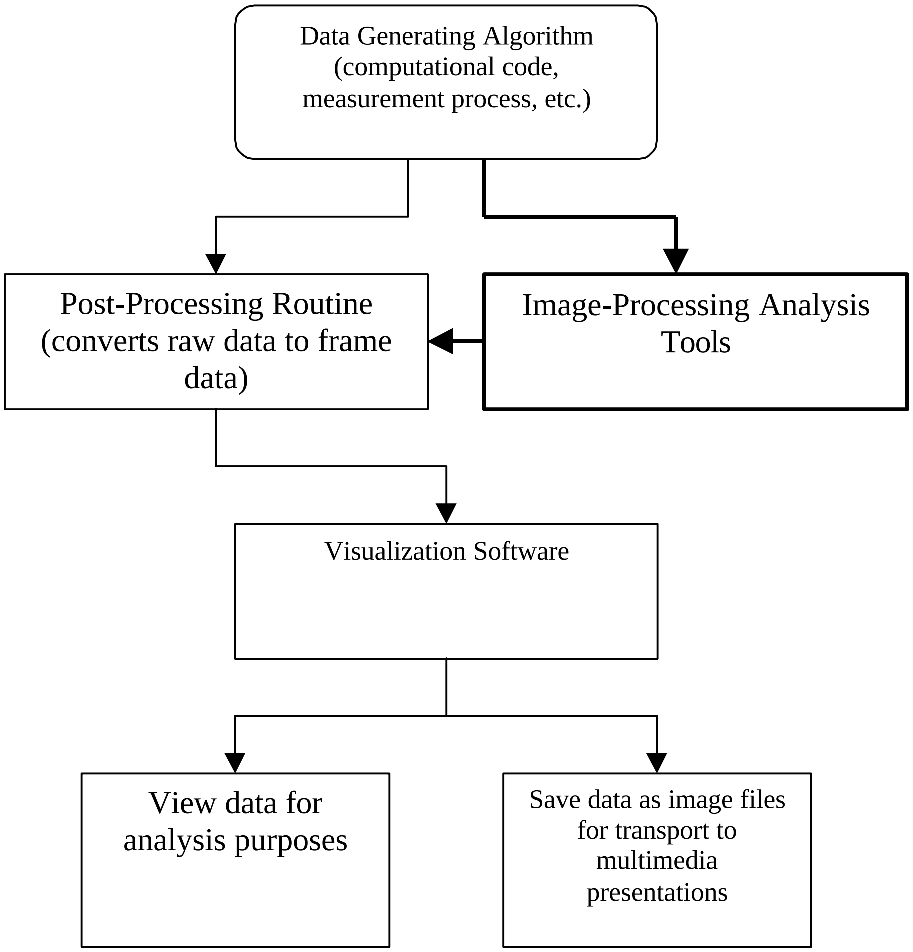

Data Generating Algorithm (computational code, measurement process, etc.)
Post-Processing Routine (converts raw data to frame data)
Image-Processing Analysis Tools
Visualization Software
View data for analysis purposes
Save data as image files for transport to multimedia presentations

Figure 4-3. Flow chart showing the essential components of visualization capability.

The raw data is provided at a constant frequency and is on a two-dimensional spatial grid. The first step in our process is to transform this data to the frequency domain using a two-dimensional Fourier transform. However, we must first check the Nyquist rate to ensure that we are not in violation of the sampling theorem. This needs to be done in each of the two dimensions. Unfortunately, if the data grid is not spaced finely enough to avoid aliasing, the FDTD model needs to be re-sampled and recomputed. Therefore, it may make sense to incorporate this check in the modeling process. As shown in Figure 4-4, the FFT transforms the data from x-y space to k_x-k_y space, where k_x and k_y are wavenumbers. In k-space we are looking at how the spectral energy is distributed amongst the various propagating modes. Areas of large amplitude correspond to modes that contain significant energy. Therefore, we can use the Gabor filter to isolate each propagating mode that is above a certain energy level.

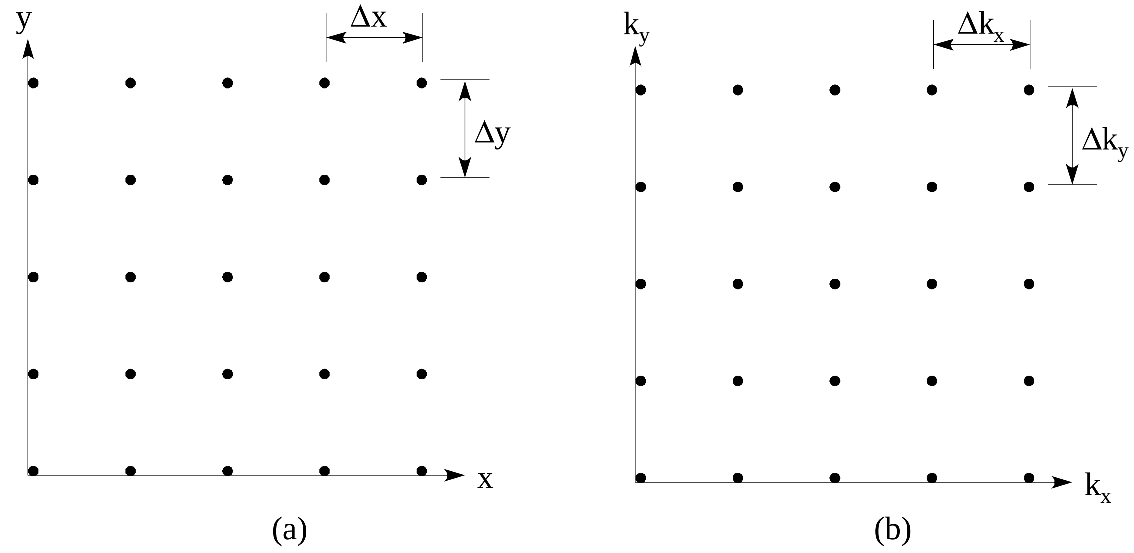


Figure 4-4. Data on the x-y plane (a) and k_x-k_y plane (b).

In order to apply the Gabor filter, we first need to decide on a window width. This will depend heavily on the spectral content of the data. If we are dealing with a resonant structure that has well defined propagation constants, then we can use a Gabor filter with a small width. On the other hand, if the propagation constants are blurred we may need to use a broader filter. Recalling that the window width is defined above as one standard deviation of the power distribution of the signal, we can set the octave bandwidth, β, of the filter using the covariance parameters as follows [21]

$$s = \frac{l_0}{2p}\left(\frac{2^b+1}{2^b-1}\right)$$
$$where\ b = \log_2\left(\frac{f_{upper}}{f_{lower}}\right). \qquad (4\text{-}10)$$

We can now sample the Gabor window function of (4-8) and multiply it with the frequency domain data. Transforming the result back to the spatial-domain using an inverse FFT will result in a data set that contains only the effects of the filtered modes. The best means of verification is to run the data through the post-processor and visualization network; this will be the topic of Chapter 5.

4.4 Strategies for Determining Wave Properties

Before we present a complicated example in Chapter 5, it might be helpful to investigate several contrived problems. This should allow us to become comfortable with analyzing spectral quantities in k-space and will also give us some practice operating our new analysis tool. Additionally, this provides an excellent means for debugging any programming errors that might exist. We will consider three examples, all of which involve time-harmonic plane waves propagating in free-space.

The first example is comprised of a uniform plane wave traveling along the x-axis. The field distribution is given by

$$E(x,y) = E_0 e^{+jk_{x0}x}, \qquad (4\text{-}11)$$

where the propagation constant, k_{x0}, is 1 rad/m and E_0 is unity. Figure 4-5 shows a grayscale contour plot of the data along with a graph of the corresponding k-space that shows the spectral components of the plane wave. There is a single spike corresponding to k_{x0}=1 rad/m and k_{y0}=0 rad/m, which is exactly what we would expect to see. This example represents a structure that supports only a single radiating mode.

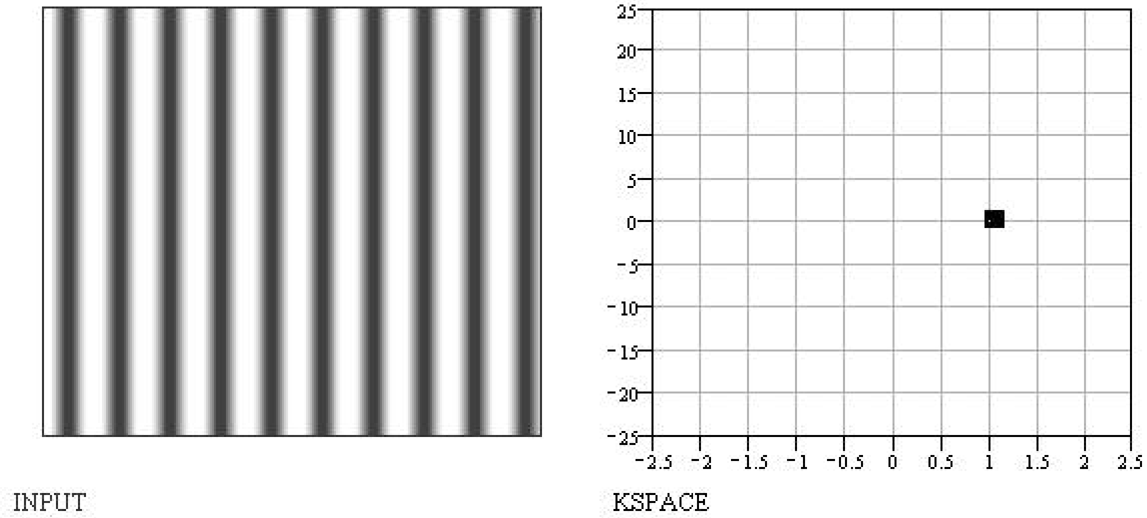
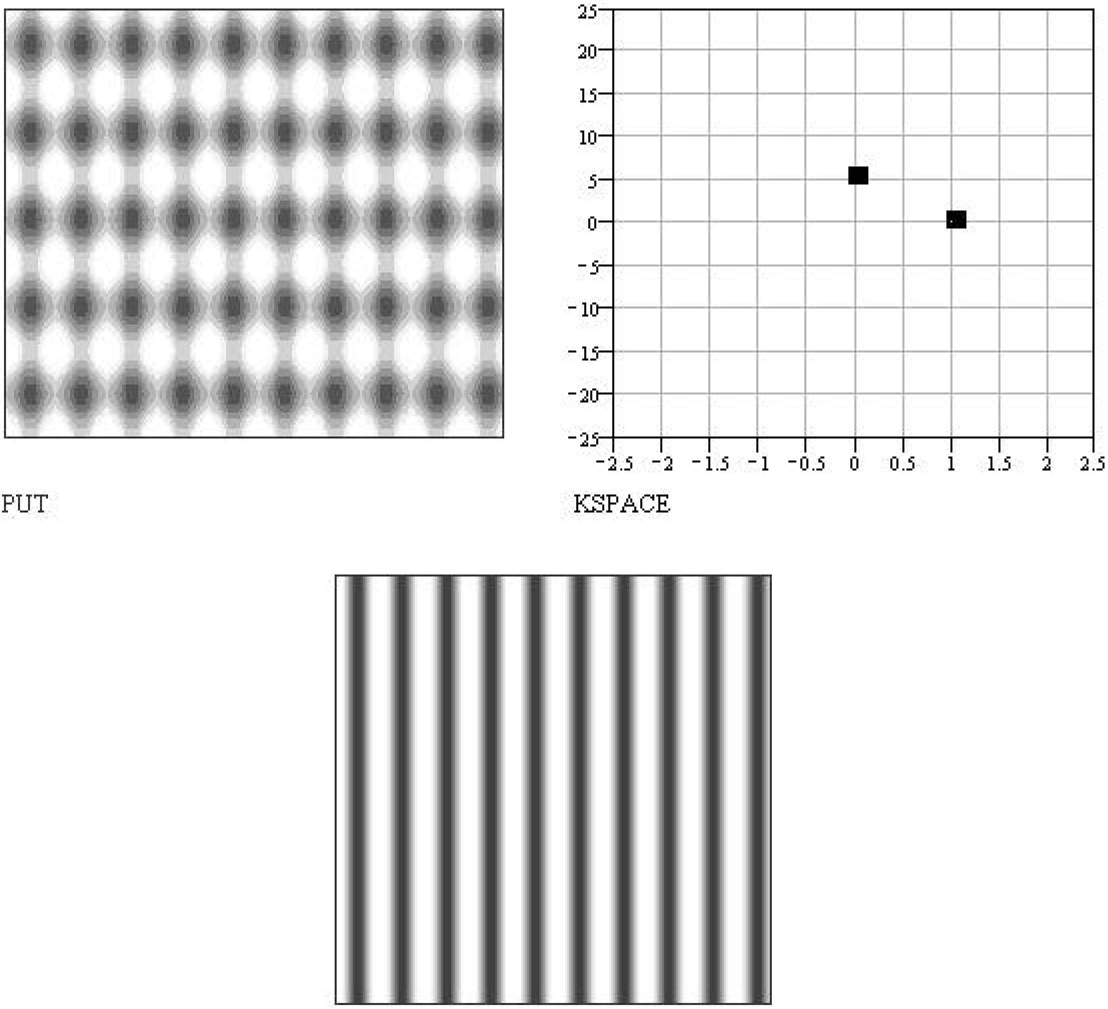


Figure 4-5. Spatial and k-space data for a uniform plane wave traveling along the x-axis.

The next example involves a superposition of two plane waves whose field equations are given by

$$E(x,y) = E_0\left(e^{+jk_{x0}x} + e^{+jk_{y0}y}\right). \qquad (4\text{-}12)$$

Here we can see that one plane wave propagates along the x-axis and the other along the y-axis. This could represent a structure that supports two different modes. In this case we elect to set k_{x0} to 1 rad/m, k_{y0} to 5 rad/m, and E_0 to 1 V/m. Figure 4-6 shows the spatial-domain data and corresponding k-space, where both spectral components behave as expected. In addition, we use the image-processing techniques to filter out the plane wave traveling along the x-axis. The result should match the data shown in Figure 4-5; and indeed it does.



INPUT

KSPACE

FILTERED

Figure 4-6. Spatial, k-space, and filtered data for a superposition of uniform plane waves given by (4-12).

The final example shows a field distribution that does not line up with either coordinate axis. The electric field is given by

$$E(x,y) = E_0 \cos\left(k_{x0}x + k_{y0}y\right), \qquad (4\text{-}13)$$

where k_{x0} is 1 rad/m, k_{y0} is 5 rad/m, and E_0 is 1 V/m. This field distribution could model the standing wave pattern that is associated with a feed line. Figure 4-7 shows the data in the spatial-domain and in k-space. Here we can see that the standing wave is simply a superposition of two traveling waves. Again, the results match what we expected.

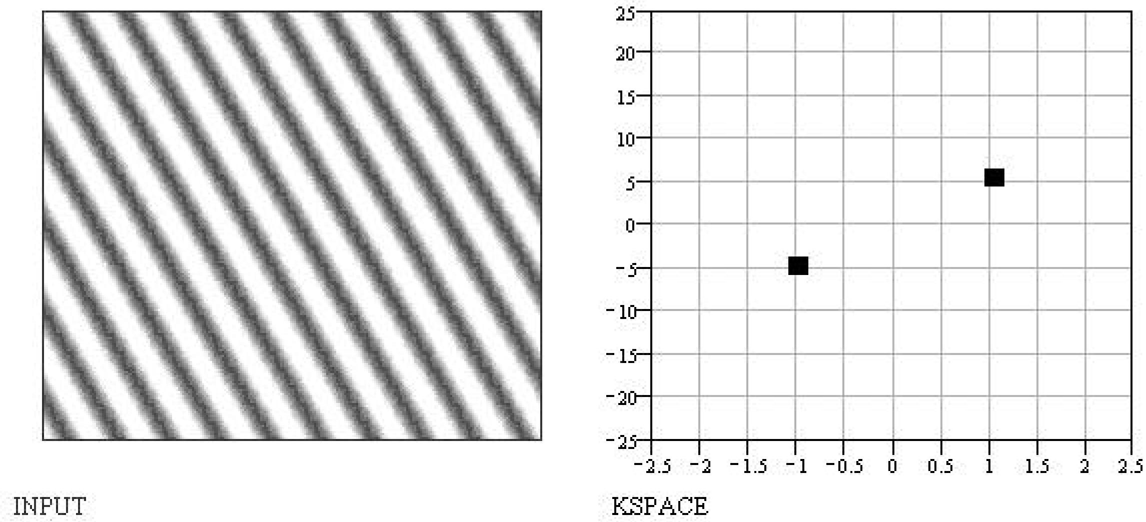


Figure 4-7. Spatial and k-space data for a uniform plane wave given by (4-13).

The above problems are intended to ease us into the more complicated example that will be presented in Chapter 5. In that example we will analyze and filter the fields of a rectangular waveguide. Instead of generating the data with mathematical functions as we did here, the FDTD code that is used throughout this thesis will be used to model and simulate the problem.