Fourier Transforms Using *Mathematica*®

Joseph W. Goodman

SPIE PRESS
Bellingham, Washington USA

Fourier Transforms Using *Mathematica*

Joseph W. Goodman
Stanford University

Preface

This book is a product of shelter-in-place. During the spring of 2020, when COVID-19 was rampant, staying inside and isolated was the recommended way to avoid infection. Under such circumstances, the mind seeks activities that will keep one occupied and stimulated. The activity of choice for me was writing a book on Fourier transforms using *Mathematica*.

Fourier transforms is a subject quite familiar to me. I had taught a graduate-level course on this subject for many years at Stanford, sometimes with more than 100 students coming from departments across the university. During this time I accumulated substantial lecture notes, upon which much of this book is based. However, this book does contain some material that was not included in the classes I taught.

Mathematica is a program I used extensively in illustrating other books, but I was by no means an expert on its capabilities. I took the combination of Fourier transforms and *Mathematica* as a challenge, and indeed I learned a great deal about this program in the course of writing this material. Still I do not consider myself a *Mathematica* expert, but rather a user who loves to find capabilities of this program that I have not yet discovered. Hopefully the reader of this book will emerge as a reasonably knowledgeable user of the program.

I learned Fourier transforms first as a graduate student in a course on the subject taught at Stanford by Ron Bracewell, a well-known radio astronomer and an innovator in many fields. In addition to his well-known books on Fourier transforms, he wrote a compendium on the Eucalyptus trees on the Stanford campus. He was a true renaissance man. His course shaped my career, and in gratitude, I am dedicating this book to his memory.

Table of Contents

1. Introduction

The Fourier transform is a ubiquitous tool used in most areas of engineering and physical sciences. The purpose of this book is two-fold: 1) to introduce the reader to the properties of Fourier transforms and their uses, and 2) to introduce the reader to the program *Mathematica* and to demonstrate its use in Fourier analysis. Unlike many other introductory treatments of the Fourier transform, this treatment will focus from the start on both one-dimensional (1D) and two-dimensional (2D) transforms, the latter of which play an important role in optics and digital image processing, as well as in many other applications. It is hoped that by the time the reader has completed this book, he or she will have a basic familiarity with both Fourier analysis and *Mathematica*.

1.1 Why *Mathematica*

Many other texts on Fourier transforms exist, especially notable among them being those by Bracewell [1], Papoulis [2], and, for a more recent book, Osgood [3]. This book differs from others in that it is devoted to exploring these topics with the help of the program *Mathematica*. The book has been written as a *Mathematica* notebook, allowing the reader to interact with the document using the free program *Wolfram Player*. Considering the various computational languages available, why have we chosen *Mathematica* here? The reasons are several. First, *Mathematica* seamlessly blends symbolic and numerical mathematics, with incredibly broad knowledge about various functions and capabilities. Second, *Mathematica* allows the creation of documents such as this, which include both mathematical computations and formatted text. Finally, if the reader has the full *Mathematica* program (rather than just *Wolfram Player*), he or she can change the various commands that are executed in this ebook and explore other functions, other parameters, etc.

Mathematica is the creation of Stephen Wolfram and the company Wolfram Research. It was first released in 1988, and has appeared in many versions. This book was written using version 12. There are many books that cover the capabilities of *Mathematica*; see for example, *Mathematica Navigator* by H. Ruskeepaa, and *The Student's Introduction to Mathematica and the Wolfram Language* by B.F. and E.A. Torrence, two among many. In addition, the Wolfram documentation available under the *Mathematica* help menu contains all you need to know about individual commands and their syntax.

Mathematica is an extremely powerful language with a huge number of commands. In general, commands have names that to some extent reveal their function. It is possible to write very dense and compact code in this language, but in this book we have chosen to sacrifice compactness for understandability. Thus the calculations presented here can often be written in more compact forms, but those forms will be harder to understand than the forms we have used here.

An attempt has been made to limit the number of *Mathematica* commands used in this book to a subset of the available commands, simply as an aid to the reader. The syntax and actions of these commands are covered in detail in the references and in the *Mathematica* Help menu, as frequency response of circuits; they are used by radio astronomers to form images from interferometric data gathered with antenna arrays; they are used by spectroscopists to obtain high-resolution spectra in the infrared from interferograms, a method known as Fourier spectroscopy; they are used by crystallographers to find crystal structure using Fourier transforms of X-ray diffraction patterns; and they are used by camera designers to specify camera performance in terms of spatial frequency response. Even psychologists have used Fourier transforms in the study of memory and perception. Almost regardless of your field, you will be able to use your knowledge of Fourier transforms to your advantage.

Back to Contents?

2. Some Useful 1D and 2D Functions

2.1 User-Defined Names for Useful Functions

Our interest here will be in both 1D and 2D functions and their transforms. As a start it is useful to define a group of 1D functions. 2D functions can be built using these functions. By a 1D function, we mean one that is defined on a line. By a 2D function, we mean one that is defined on a plane. The following table provides a set of 1D functions that have user-defined names (all beginning with lower-case letters) and the corresponding *Mathematica* code that defines those functions. Again, these definitions are included in an initialization cell near the beginning of this document.

```
rect[x_] := UnitBox[x];
sinc[x_] := Sinc[Pi * x];
step[x_] := UnitStep[x];
sgn[x_] := Sign[x];
gaus[x_] := Exp[-Pi * x^2];
jinc[x_] := 2 * BesselJ[1, Pi * x] / (Pi * x);
tri[x_] := UnitTriangle[x];
rtri[x_] := rect[x - 1 / 2] * UnitTriangle[x];
δ[x_] := DiracDelta[x];
comb[x_] := DiracComb[x];
```

A table showing plots of most of these functions follows. The functions $\delta(x)$ and comb(x) will be discussed in Section 2.5. Note that the command to plot such functions is

```
Plot[g[x], {x, lower, upper}, PlotRange → All],
```

where `lower` and `upper` are the lower and upper limits, respectively, for which the function should be plotted.

and in two dimensions

$$\int\int_{-\infty}^{\infty} g(x,\, y)\,\text{comb}(x,\, y)\, dx\, dy = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty} g(n,\, m).$$

An important important property is that $\text{comb}(a x) = \frac{1}{|a|}\text{comb}\left(\frac{x}{a}\right)$.

Back to Contents?

3. Definition of the Continuous Fourier Transform

3.1 The 1D Fourier Transform and Inverse Transform

We begin with a discussion of 1D Fourier transforms. We start with a function $g(x)$ defined on the x axis, and calculate its 1D transform $\mathcal{G}(u)$ defined on the u axis. There are several possible definitions of the Fourier transform, differing through scaling constants applied to the transform and/or to its argument. *Mathematica* defines the 1D Fourier transform $\mathcal{G}(u)$ of the function $g(x)$ through the equation

$$\mathcal{G}(u) = \sqrt{\frac{|b|}{(2\pi)^{1-a}}}\int_{-\infty}^{\infty} g(x)\exp(i\, b\, u\, x)\, dx,$$

where a and b are the scaling constants to be chosen. The symbol i represents the imaginary constant $\sqrt{-1}$, while a and b are called the *Fourier parameters*. The symbol x may represent time, in which case the symbol u represents temporal frequency, measured in hertz (Hz).

The definition of the transform to be used here and throughout is one that chooses the constants $(a, b) = (0, -2\pi)$, yielding

$$\mathcal{G}(u) = \int_{-\infty}^{\infty} g(x)\exp(-i\, 2\,\pi\, u\, x)\, dx.$$

This equation now has to be expressed in *Mathematica* language. The expression that performs this operation on an input function g(x) is

```
𝒢[u_] := FourierTransform[g[x], x, u, FourierParameters → {0, -2 * Pi}]
```

Notice the underbar following u in the definition of the function $\mathcal{G}$, as is required in any definition of a function in *Mathematica.* Notice also the delayed equality symbol **:=**, which means that the expression will not be evaluated until it is needed in later code.

Paralleling the definition of the Fourier transform is that of the *inverse* Fourier transform. *Mathematica*'s definition of this quantity is

$$g(x) = \sqrt{\frac{|b|}{(2\pi)^{1+a}}}\int_{-\infty}^{\infty}\mathcal{G}(u)\exp\left(-i\, b\, u\, x\right) du.$$

To return from $\mathcal{G}$ to g given our definition of the transform requires that we again choose the Fourier parameters for the inverse transform as $\{a, b\} = \{0, -2\pi\}$. We then obtain

$$g(x) = \int_{-\infty}^{\infty}\mathcal{G}(u)\exp(i\, 2\,\pi\, u\, x)\, du.$$

This equation can be thought of as expressing $g(x)$ as a superposition of an infinite number of weighted complex exponentials of the form $\mathcal{G}(u)$ $\exp(i\ 2\ \pi\ u\ x)$, where $\mathcal{G}(u)$ is the weighting function for the complex exponential with frequency u. The *Mathematica* command to perform the inverse Fourier transform is given by

```
g[x_] := InverseFourierTransform[𝒢[u], u, x, FourierParameters -> {0, -2 * Pi}]
```

For all but the most unusual functions, the inverse Fourier transform of the Fourier transform of a function $g(x)$ returns the function $g(x)$, except at points of discontinuity of g, where the average of the values of g from the right and from the left of the discontinuity is obtained. Discussion of the conditions for successful inversion of the Fourier transform by the inverse Fourier transform will follow later.

3.2 The 2D Fourier Transform and Inverse Transform

In this text, we shall often consider 2D functions that are defined on an $(x,\ y)$ spatial plane. The Fourier transforms of such functions are likewise 2D functions defined on a frequency plane. Again there are several different definitions of the 2D transform that differ from one another through scaling of the transform or the frequency variables, or both. *Mathematica* defines the 2D Fourier transform through the equation

Back to Contents?

4. Convolutions and Correlations

In many fields of physics and engineering, the convolution integral and the correlation integral play important roles in analysis. In this chapter we explore these integrals in one and two dimensions, and discuss their implementations in *Mathematica*.

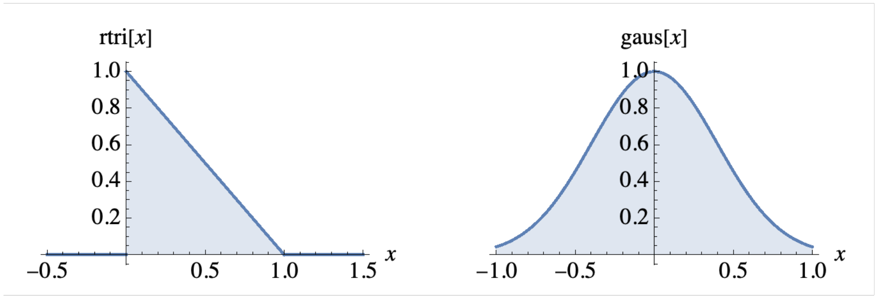
4.1 Convolution Integrals

The convolution $g(x)$ between two functions $h(x)$ and $f(x)$ is defined by the integral

$$g(x) = \int_{-\infty}^{\infty} f(\xi)\, h(x-\xi)\, d\xi = f(x) * h(x).$$

Here the symbol ξ is simply a dummy variable of integration, and the symbol $*$ stands for convolution between the two stated functions. (Do not confuse this symbol with the multiplication symbol used in *Mathematica* code.) Note that one of the functions, in this case $h(\xi)$, has been reversed left to right and shifted to be centered at $\xi = x$. The same resulting $g(x)$ is obtained if $f(\xi)$, rather than $h(\xi)$, is reversed and shifted.

Consider the case of functions $f(x) = \text{gaus}(x)$ and $h(\xi) = \text{rtri}(x)$. Both functions are shown below:



Visualizing the area under the convolution integral can be accomplished with the *Mathematica* command **`Manipulate`**. The form of this command is

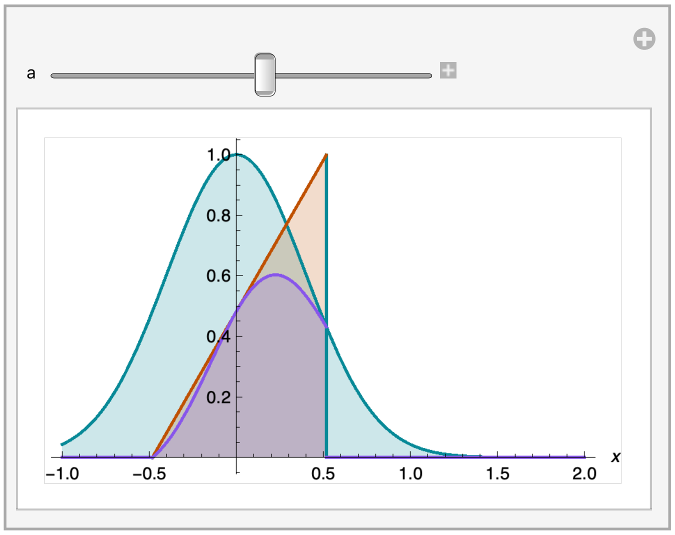
```
Manipulate[expression, {a, amin, amax}].
```

Note that to use this command, you must enable "Dynamic Updating" in the Evaluation menu. If the resulting plot shows only an outline of the coordinate system, highlight the cell marker for the command on the right and execute the command by pressing shift-return, assuming you are using the full *Mathematica* program.

This expression allows one, for example, to plot a function with a variable parameter, and to visualize the changes in the function as the parameter is varied. In the present case, the command needed is

```
In[∘]:= Manipulate[Plot[{gaus[x], rtri[-x + a], gaus[x] * rtri[-x + a]}, {x, -1, 2},
          PlotRange → All, ExclusionsStyle → Automatic, AxesLabel → {x, None},
          Filling → Axis, FillingStyle → Automatic], {a, -0.9, 1.6}, SaveDefinitions → True]
```

Out[∘]=



In this plot, axis labels are explicitly called for, and the curves are colored down to the *x* axis. By moving the button above the plot, the overlap of the two functions being convolved changes, and the reader can see the integrand of the convolution integral plotted.

Mathematica has a command for convolving two functions *f* and *h*:

```
g[x_] = Convolve[f[ξ], h[ξ], ξ, x].
```

As an example, consider the convolution of the right half-triangle with a scaled version of itself:

```
In[∘]:= g[x_]  = Convolve[rtri[2 * ξ], rtri[ξ], ξ, x]
```

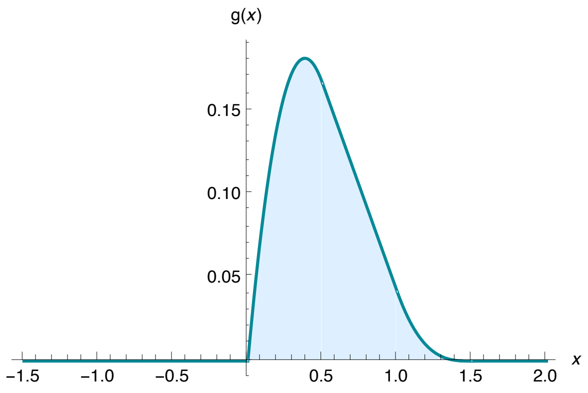
Out[∘]=

$$\begin{cases} \frac{1}{24}(7-6x) & \frac{1}{2} < x < 1 \\ -\frac{1}{24}(-3+2x)^3 & 1 \le x < \frac{3}{2} \\ \frac{1}{6}x\left(6-9x+2x^2\right) & 0 < x \le \frac{1}{2} \\ 0 & \text{True} \end{cases}$$

The result is quite complex, but it can be visualized with a plot:

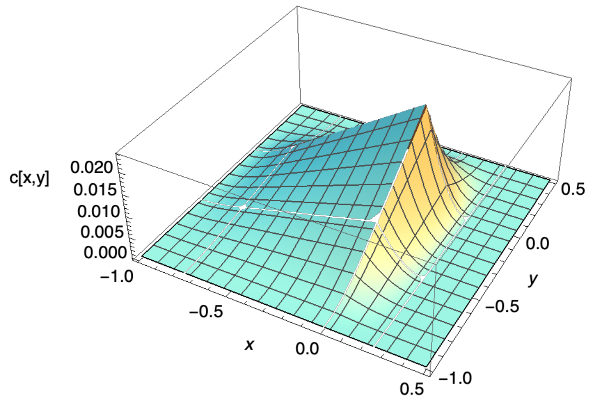
```
In[∘]:= Plot[g[x], {x, -1.5, 2.0}, AxesLabel → {x, "g(x)"}]
```

Out[∘]=



```
In[•]:= Plot3D[c[x, y], {x, -1, 0.5}, {y, -1, 0.5},
          PlotRange → All, PlotPoints → 40, AxesLabel → {x, y, "c[x,y]   "}]
```

Out[•]=



You should be aware that in some cases *Mathematica* is unable to find the convolution or the correlation. For example,

```
In[•]:= c[x_] = Convolve[jinc[ξ], jinc[-ξ], ξ, x]
```

$$\text{Out[•]=}\quad \frac{4\,\text{Convolve}\left[\frac{\text{BesselJ}[1,\pi\,\xi]}{\xi},\ \frac{\text{BesselJ}[1,\pi\,\xi]}{\xi},\ \xi,\ x\right]}{\pi^2}$$

We note that it is often still possible to find the correlation function by reasoning in the frequency domain using the autocorrelation theorem described in the Section 5.4.

Back to Contents?

5. Some Useful Properties of Fourier Transforms

Fourier transforms have a number of interesting properties that are of great help in analyzing problems. In this chapter we discuss some of these properties in detail. Our attention is limited to properties for which *Mathematica* is helpful or relevant.

5.1 Symmetry Properties of Fourier Transforms

Symmetry properties in the x domain result in other symmetry properties in the u domain; we explore these properties in this section.

Every function $g(x)$ can be written as the sum of an even part $e(x)$ and an odd part $o(x)$, where

$$g(x) = e(x) + o(x), \qquad e(x) = \frac{g(x) + g(-x)}{2}, \qquad o(x) = \frac{g(x) - g(-x)}{2}$$

and this decomposition is unique. However, the decomposition does depend on the origin chosen for $g(x)$. For example, $\cos(x)$ is even, but $\cos(x - \pi/2) = \sin(x)$ is odd.

Consider now the symmetry properties of the Fourier transform $\mathcal{G}(u)$ that result when we impose symmetry properties on $g(x)$. The Fourier transform of $g(x)$ can be written as

$$\mathcal{G}(u) = \int_{-\infty}^{\infty} (e(x) + o(x))\,(\cos(2\,\pi\,u\,x) - i\sin(2\,\pi\,u\,x))\,dx =$$

$$\int_{-\infty}^{\infty} e(x)\cos(2\,\pi\,u\ x)\,dx - i\int_{-\infty}^{\infty} e(x)\sin(2\,\pi\,u\,x)\,dx + \int_{-\infty}^{\infty} o(x)\cos(2\,\pi\,u\,x)\,dx - i\int_{-\infty}^{\infty} o(x)\sin(2\,\pi\,u\,x)\,dx,$$

where we have expanded the complex exponential into a sum of a real cosine term and an imaginary sine term.

The infinite integral of any function that is odd in x will vanish. Note that $e(x)\sin(2\pi u x)$ and $o(x)\cos(2\pi u x)$ are both odd functions in x, and therefore their integrals vanish, leaving two integrals with integrands that are even functions of x,

$$\mathcal{G}(u) = \int_{-\infty}^{\infty} e(x)\cos(2\pi u x)\,dx - i\int_{-\infty}^{\infty} o(x)\sin(2\pi u x)\,dx.$$

From this set of two integrals we can determine the properties of the even and odd parts of $\mathcal{G}(u)$ by focusing on their u dependencies. The table below shows the resulting symmetry properties of $\mathcal{G}(u)$:

$g(x)$	$\mathcal{G}(u)$
even and real	even and real
even and imaginary	even and imaginary
odd and real	odd and imaginary
odd and imaginary	odd and real

We conclude from this table that when $g(x)$ is real, we have $\mathcal{G}(-u) = \mathcal{G}^*(u)$ and we say that $\mathcal{G}(u)$ has *Hermitian symmetry*, while when $g(x)$ is imaginary, $\mathcal{G}(-u) = -\mathcal{G}^*(u)$ and we say that $\mathcal{G}(u)$ has *anti-Hermitian* symmetry. Note also that when $g(x)$ is even, the symmetry properties of $\mathcal{G}(u)$ are the same as the symmetry properties of $g(x)$, while when $g(x)$ is odd, they are not the same.

We illustrate these properties with two examples: $g(x) = \text{tri}(x)*\text{sgn}(x)$, an odd and real function, and $i\,\text{tri}(x)$, an even and imaginary function:

```
In[∘]:= ft1[tri[x] * sgn[x]]
```

$$\text{Out[∘]=}\quad \frac{i\,(-2\pi u + \text{Sin}[2\pi u])}{2\pi^2 u^2}$$

```
In[∘]:= ft1[I * tri[x]]
```

$$\text{Out[∘]=}\quad i\,\text{Sinc}[\pi u]^2$$

In the first example, we see that the transform is odd and imaginary, as predicted, while in the second example the transform is even and imaginary, also as predicted.

5.2 Area and Moment Properties of 1D Fourier Transforms

There is an intimate relationship between moments of functions in the x domain and the behavior of their Fourier transforms at the origin in the frequency domain. The simplest of these relations is an "area" property (the zeroth-order moment). According to this property,

$$\int_{-\infty}^{\infty} g(x)\,dx = \lim_{u\to 0}\int_{-\infty}^{\infty} g(x)\,e^{-i2\pi u x}\,dx = \mathcal{G}(0).$$

To verify this theorem using *Mathematica*, we first integrate the Gaussian function to find its area, and then calculate the value of its Fourier transform at the origin. The two commands follow.

```
In[∘]:= Integrate[gaus[x], {x, -Infinity, Infinity}]

Out[∘]= 1

In[∘]:= Limit[ft1[gaus[x]], u → 0]

Out[∘]= 1
```

The equality of these two quantities holds for any function that is absolutely integrable. It also holds for most transforms in-the-limit.

There exist related correspondences between moments of functions in the x domain and derivatives of their Fourier transforms at the origin in the frequency domain. Assuming that

$$\int_{-\infty}^{\infty} \left| x^k\, g(x) \right| dx < \infty,$$

starting with the frequency domain expression and moving to the x domain,

$$\left(\frac{i}{2\pi}\right)^k \underset{u\to 0}{\text{Lim}}\, \frac{d^k}{du^k}\left[\int_{-\infty}^{\infty} g(x)\,e^{-i2\pi u x}\,dx\right.$$

$$= \left(\frac{i}{2\pi}\right)^k \underset{u\to 0}{\text{Lim}}\left[\int_{-\infty}^{\infty} g(x)\,\frac{d^k}{du^k}e^{-i2\pi u x}\,dx\right] = \left(\frac{i}{2\pi}\right)^k \underset{u\to 0}{\text{Lim}}\left[\int_{-\infty}^{\infty}(-i2\pi x)^k\, g(x)\,e^{-i2\pi u x}\,dx\right. = \int_{-\infty}^{\infty} x^k\, g(x)\,dx.$$

Thus,

The following examples of the first, second, and third derivatives of gaus[*x*] may be helpful:

First derivative:

```
In[•]:= Clear[𝒢1, 𝒢2, 𝒢3]
        𝒢1[u_] = ft1[D[gaus[x], {x, 1}]]
```

Out[•]= $2\,i\,e^{-\pi u^2}\,\pi\,u$

Second derivative:

```
In[•]:= 𝒢2[u_] = ft1[D[gaus[x], {x, 2}]]
```

Out[•]= $-4\,e^{-\pi u^2}\,\pi^2\,u^2$

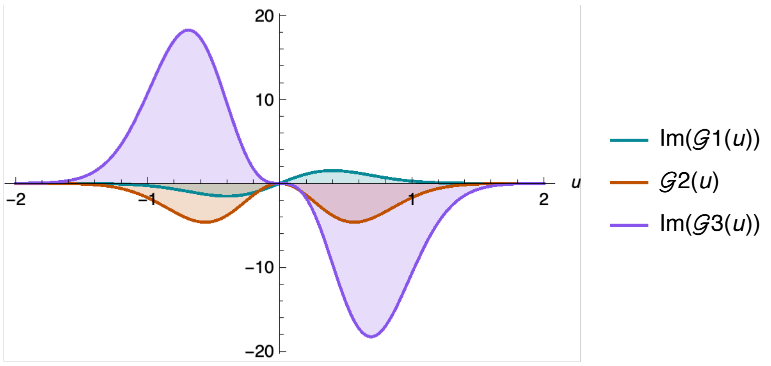
Third derivative:

```
In[•]:= 𝒢3[u_] = ft1[D[gaus[x], {x, 3}]]
```

Out[•]= $-8\,i\,e^{-\pi u^2}\,\pi^3\,u^3$

You can see that in each case the spectrum is indeed as predicted above. Note that for every odd derivative, the result for a real $\mathcal{G}(u)$ is imaginary, while for every even derivative, the result for a real $\mathcal{G}(u)$ is real-valued.

```
In[•]:= Plot[{Im[𝒢1[u]], 𝒢2[u], Im[𝒢3[u]]}, {u, -2, 2}, PlotRange → All,
          PlotLegends → "Expressions", FillingStyle → Automatic, AxesLabel → {u, None}]
```

Out[•]=


The command `PlotLegends->Automatic` plots a legend next to the figure. Note that *Mathematica* can even find an expression for the *n*th derivative of the Gaussian function, although it takes some time to compute:

```
In[•]:= ft1[D[gaus[x], {x, n}]]
```

Out[•]= $2^{-1+n}\,e^{\frac{i n \pi}{2}-\pi u^2}\left(-\frac{1}{\pi}\right)^{-n}\text{Abs}[u]^{-1+n}$
$\left(-u+\text{Abs}[u]+\left(u+\text{Abs}[u]\right)\text{Cos}[n\,\pi]+\left(u+\text{Abs}[u]\right)\text{Erfi}\left[\sqrt{\pi}\,\text{Abs}[u]\right]\text{Sin}[n\,\pi]\right)$

In this result, the `Erfi` function is the so-called "imaginary error function," described in the *Mathematica* Help menu under Wolfram documentation.

The derivative theorem also works in reverse, i.e., for derivatives in the frequency domain. The relationship in this reverse direction is

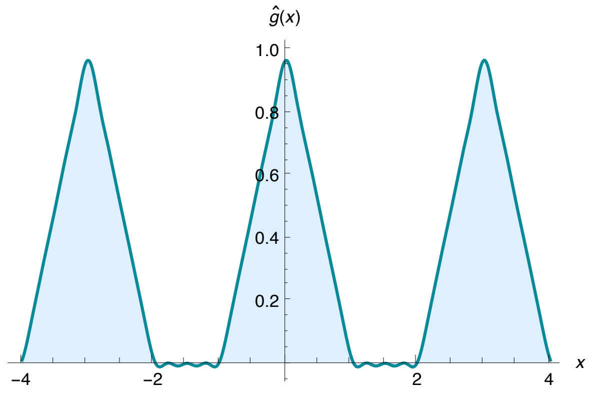
$$\frac{d^k}{d u^k}\mathcal{G}(u) = (-i 2\pi\, x)^k\, g(x).$$

5.5 The Projection-Slice Theorem

The projection-slice theorem is sufficiently important to warrant its own section. This theorem states that given a projection (to be defined below) through a 2D function $g(x, y)$ at angle θ to the x axis, the 1D Fourier transform of that projection is a central slice through the 2D spectrum of $g(x, y)$ at angle θ to the u axis in the frequency domain. Projections are at the heart of computerized tomography systems used in medical X-ray and MRI machines.

```
In[•]:= Clear[c, d, e]
        c[n_] = (1 / 3) * ft1[tri[x] * rect[x / 3]] /. u → n / 3;
        Quiet[d = Table[c[n], {n, -10, 10}]];
        d[[11]] = Limit[c[n], n → 0];
        e = Table[Exp[I * 2 * Pi * (n / 3) * x], {n, -10, 10}];
        Plot[d.e, {x, -4, 4}, Filling → Axis, AxesLabel → {x, "ĝ(x)"}]
```

Out[•]=



This has been our first exposure to lists in *Mathematica*. We will encounter them in much more detail when we consider the discrete Fourier transform in the next chapter.

9.3 *Mathematica* Commands for Fourier Series

In the section above, we have shown the relations between Fourier transforms and complex Fourier series, including methods for constructing such series given one period of a periodic function. *Mathematica* actually has a set of commands that make these calculations much easier. In what follows, we keep the same definition of the function $p(x)$ defining one period, i.e., $p(x) = \text{tri}(x)\,\text{rect}(x/3)$.

The most useful *Mathematica* command with respect to complex Fourier series is the `FourierSeries[p[x],x,n,FourierParameters->{a,b}]` command. *Mathematica* defines this command to find Fourier coefficients c_k according to

$$c_k = \left|\frac{b}{2\pi}\right|^{\frac{a+1}{2}} \int_{-\frac{\pi}{|b|}}^{\frac{\pi}{|b|}} g(x)\, e^{-i\,b\,k\,x}\, dx, \qquad k = -n,\ \ldots,\ n.$$

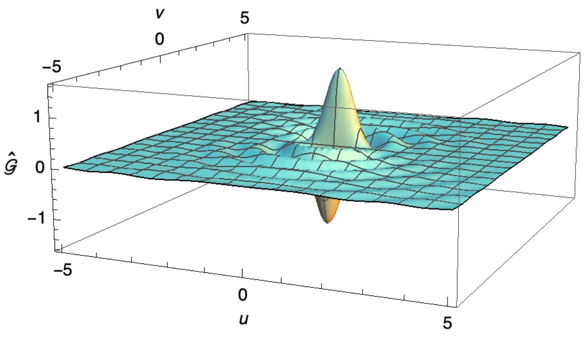
To have this representation of the c_k conform to those derived above, the Fourier parameters (a, b) must be chosen as `{1,2*Pi/L}`, where `L` is the period of the periodic function. We illustrate with the same periodic function used above, for which the period is 3. We choose $n = 4$, as we did in the first case above.

```
In[•]:= FullSimplify[f.e]
```

Out[•]= $\frac{2}{3}\pi^2\rho\,\text{HypergeometricPFQ}\left[\left\{\frac{3}{2}\right\},\left\{2,\frac{5}{2}\right\},-\pi^2\rho^2\right]\text{Sin}[\phi]$

We can see this result by plotting it. But first we convert it to a function of rectangular coordinates using the command `/.{ρ→Sqrt[u^2+v^2],ϕ→ArcTan[u,v]}`. The plot is then:

```
In[•]:= Plot3D[f.e /. {ρ → Sqrt[u^2 + v^2], ϕ → ArcTan[u, v]}, {u, -5, 5},
          {v, -5, 5}, AxesLabel → {u, v, "𝒢̂"}, PlotRange → All, PlotPoints → 30]
```

Out[•]=


Thus, the cos(θ) variation of the phase and the circ(r) dependence on radius in the (r, θ) plane have created a Fourier transform with a positive mound and a negative mound, due to the sin(ϕ) dependence in the transform plane.

Note that, while we have illustrated with a simple function that is separable in polar coordinates, the method also works when the Fourier coefficients of the phase depend on r, although the computation times can be long.

We turn now to the discrete Fourier transform.

Back to Contents?

10. The Discrete Fourier Transform

When we are dealing with discrete data, Fourier analysis plays an important role, for both 1D and 2D data. However, the Fourier transform must be modified to accommodate discrete data rather than continuous data. These modifications are the subject of this chapter.

10.1 Sampling in Both Domains

Suppose we wish to estimate the Fourier transform of a physical process $f(x)$ that we imagine has existed from $x = -\infty$ and will continue to exist in the future for all x. We can measure $f(x)$ only over a limited duration of x, and therefore must base our estimate of the spectrum on a finite segment,

$$g(x) = \begin{cases} f(x) & 0 \le x < L \\ 0 & \text{otherwise.} \end{cases}$$

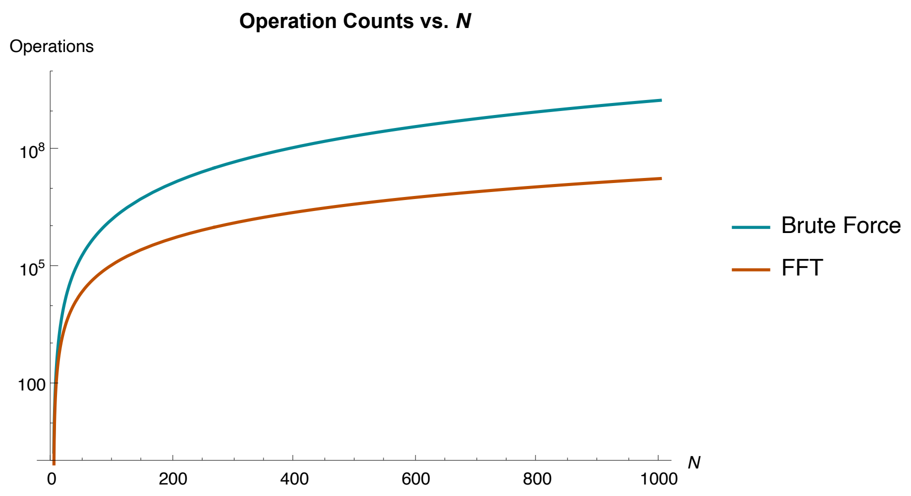
Our measurement instruments can only measure samples of $g(x)$, so we must base our spectral estimate on a collection of such samples. While a spectrum of a finite duration signal can not be bandlimited, it can be approximately bandlimited to a total bandwidth B, so we can sample with spacing $\Delta x = 1/B$ with minimum aliasing in the spectral domain. As we have seen, the result of this sampling is to replicate the spectrum to create a periodic Fourier transform, with a period given by the reciprocal of the sampling interval, i.e., a period of width B or greater if we sample in the x domain with spacing $1/B$ or smaller.

It is not generally possible to calculate a continuous spectrum of $g(x)$, for it would involve computation of an infinite number of spectral samples spaced infinitesimally apart, so we must also sample the spectrum. Since the data sample is limited to duration L, we can sample the spectrum with spacing $\Delta u = 1/L$ (or finer) without incurring aliasing in the x domain. If we sample in the x domain with the minimum allowable spacing, the number of samples of $g(x)$ will be $N = L/(1/B) = LB$.

Thus, when dealing with real data, we must work with finite sets of samples in both the x domain and the u domain. The discrete Fourier transform (DFT) is the tool that allows such spectral estimation to be performed. In this chapter we explore the DFT and its properties.

```
In[•]:= LogPlot[{2 * n^3, 2 * n^2 * Log[2, n]}, {n, 1, 1000},
          PlotRange → {10^0, 10^10}, AxesLabel → {N, "Operations"},
          PlotLegends → {"Brute Force", "FFT"}, PlotLabel → "Operation Counts vs. N"]
```

Out[•]=



For a 1000 × 1000 image array, the reduction in operations when performing the FFT rather than the brute force DFT is about a factor of 100. This reduction often makes the difference between an uncomfortably long computation and a much more reasonable computation time.

Back to Contents?

11. The Fresnel Transform

A close relative of the Fourier transform is the *Fresnel* transform, a mathematical operation widely used in signal processing, optics, acoustics, and electromagnetics. In this chapter we explore the Fresnel transform for both continuous functions and discrete data. The 1D Fresnel transform is important in radar signal theory. The 2D Fresnel transform is important in the theory of diffraction of waves of various types.

11.1 Definition of the 1D Fresnel Transform

The definitions of the 1D Fresnel transform differ in signal processing and in optics. For signal processing, the 1D Fresnel transform $f(u)$ of a function $g(x)$ is defined by

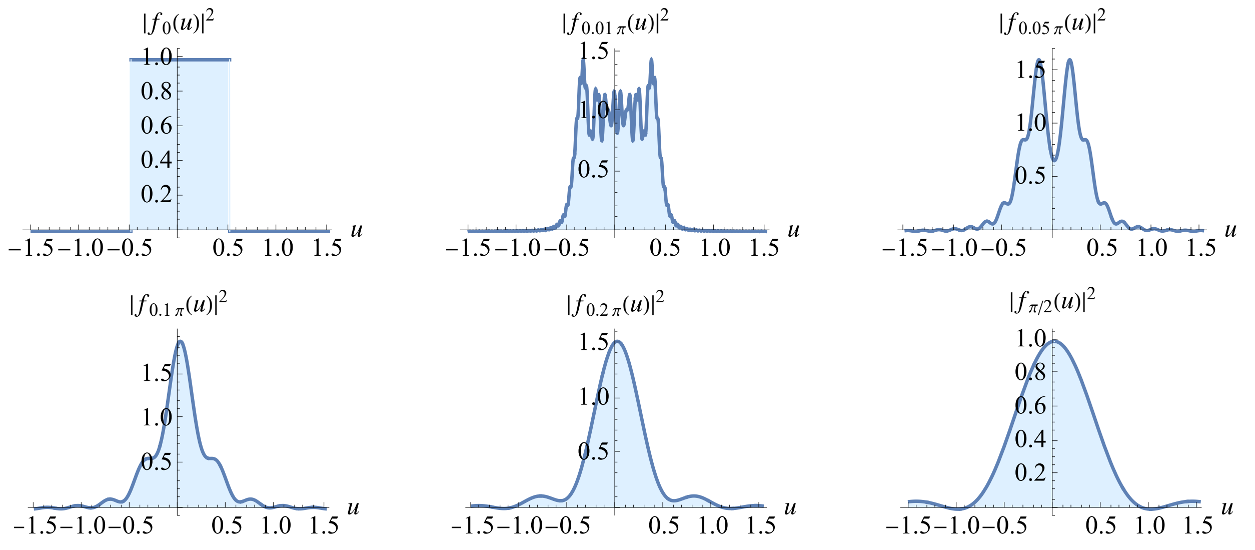
$$f(u) = \int_{-\infty}^{\infty} g\,(x)\, e^{i\pi a x^2}\, e^{-i2\pi x u}\, d\xi.$$

That is, the Fresnel transform is the Fourier transform of the product of a function $g(x)$ and $e^{i\pi a x^2}$. The constant a is real-valued and has dimensions 1/(dimension of $x)^2$. Since the operation is a Fourier transform, all Fourier transform theorems apply. The 1D exponential with argument proportional to x^2 that appears in the equation is referred to as a "quadratic-phase exponential" function.

In one dimension, $g(x)$ is often bounded to a finite interval $\pm\, L/2$. The finite bound can be included in the definition of the function $g(x)$. Note that the Fresnel transform becomes a Fourier transform of $g(x)$ when a is small enough for the quadratic-phase exponential to be ignored. We defer the definition of the Fresnel transform used in optics to Section 11.4.

11.2 Approximations to the Bandwidth of the Interval-Limited Quadratic-Phase Exponential

The width of the Fresnel transform $f(u)$ is related to the bandwidth of the interval-limited quadratic-phase exponential function. By an interval-limited function, we mean one that is non-zero over only a finite interval, $(-L/2\,,\,L/2)$. Since the Fresnel transform is the Fourier transform of a product of two functions, its width must be the width of a convolution of the Fourier transforms of $g(x)$ and $e^{i\pi a x^2}$. If we wish to know the sampling interval required for this signal, we need to know its bandwidth. The first approach to finding that width is to actually apply a Fourier transform to the function without an interval limitation. Thus,



As can be seen, the resulting distributions have shapes similar to the shapes of Fresnel diffraction patterns of a rectangular aperture measured with different values of λz. In the next section we explore the differences between the fractional Fourier transform and the Fresnel diffraction integral.

12.3 Relationship Between the Fractional Fourier Transform and the Fresnel Diffraction Integral

To facilitate a direct comparison between the Fresnel diffraction integral and the fractional Fourier transform, we present both forms again:

$$f(x) = \frac{1}{\sqrt{i\,\lambda\,z}}\, e^{i\,\frac{\pi}{\lambda z}\,x^2} \int_{-\infty}^{\infty} g(\xi)\, e^{i\,\frac{\pi}{\lambda z}\,\xi^2}\, e^{-i\,\frac{2\pi}{\lambda z}\,\xi\,x}\, d\xi$$

$$f_\alpha(u) = \sqrt{1 - i\cot(\alpha)}\; e^{i\,\pi\cot(\alpha)\,u^2} \int_{-\infty}^{\infty} e^{-i\,2\,\pi\left(\csc(\alpha)\,u\,x - \frac{\cot(\alpha)}{2}\,x^2\right)}\, g(x)\, dx.$$

The first observation concerns dimensionality. In the Fresnel diffraction integral, $f(x)$ and $g(x)$ have the same dimensions. This is not the case for the fractional Fourier transform. $f_\alpha(x)$ has dimension given by the product of the dimension of g and the dimension of x.

Another major difference between the two forms concerns scaling factors. It is clear from a comparison of the quadratic-phase exponentials that, for these factors, $\cot(\alpha)$ in the fractional Fourier transform plays the same role as $1/\lambda z$ in the Fresnel diffraction integral. On the other hand, if we compare the linear-phase exponentials, we see that $\csc(\alpha)$ is playing the same role as $1/\lambda z$. $\cot(\alpha)$ and $\csc(\alpha)$ are not the same, so we conclude that the fractional Fourier transform has a different scaling factor than the Fresnel diffraction integral. The scaling factor for the quadratic-phase exponentials is $\cot(\alpha) = \cos(\alpha)/\sin(\alpha)$, while the scaling factor for the linear-phase exponential is $\csc(\alpha) = 1/\sin(\alpha)$, which explains why the patterns of $|f_\alpha(x)|^2$ do not expand as fast with increases in $\csc(\alpha)$ as they do with increases of λz in the Fresnel diffraction integral.

Finally, there are different scaling factors in front of the integrals in the two equations. As a consequence, the heights of $|f_\alpha(x)|^2$ in the fractional Fourier transform results do not fall as fast as the heights of $|f(x)|^2$ in the Fresnel diffraction integral results.

Back to Contents?

13. Other Transforms Related to the Fourier Transform

13.1 The Abel Transform

We have discussed the projection-transform method for finding a radial slice through the Fourier transform of a given function. When that

Here `Gamma[s]` represents the gamma function with argument *s*.

To demonstrate Mellin inversion,

```
In[•]:= InverseMellinTransform[b^-s Gamma[s], s, x]

Out[•]= e^-b x
```

The Mellin transform also has a 2D version, as illustrated by

```
In[•]:= Assuming[b ∈ Reals && c ∈ Reals, MellinTransform[rtri[x / b] * gaus[y / c], {x, y}, {sx, sy}]]
```

$$\text{Out[•]=}\begin{cases} \dfrac{b^{sx}\left(\frac{1}{c^2}\right)^{-sy/2}\pi^{-sy/2}\,\text{Gamma}\left[\frac{sy}{2}\right]}{2\,sx+2\,sx^2} & b>0 \\ 0 & \text{True}\end{cases}$$

The Fourier–Mellin Transform

If the region of convergence of the Mellin transform includes the $s = i\omega$ axis, then by substituting $i\omega$ for s, we obtain what has been called the Fourier–Mellin transform. There is a property of this transform that is of particular interest in pattern recognition. The reason for this interest is illustrated by the following Fourier–Mellin examples:

```
In[•]:= FullSimplify[Abs[MellinTransform[rect[x], x, s]] /. s → I * ω]
```

$$\text{Out[•]=}\ \frac{2^{\text{Im}[\omega]}}{\text{Abs}[\omega]}$$

We have taken the absolute values of the Mellin transform in this and the following case. Note that Im[ω] = 0, so the numerator of this fraction is unity, and the result is

$$|\mathcal{M}(i\omega)| = \frac{1}{|\omega|}.$$

Now consider a magnified or de-magnified version of the same function, with the constant $b > 0$:

```
In[•]:= FullSimplify[Abs[MellinTransform[rect[x / b], x, s]] /. s → I * ω]
```

$$\text{Out[•]=}\ \frac{2^{\text{Im}[\omega]}\,e^{\text{Im}\left[\omega\,\text{Log}\left[\frac{1}{b}\right]\right]}}{\text{Abs}[\omega]}$$

Since both ω and b are real and because $b > 0$, both terms in the numerator are unity and the resulting absolute value of the Fourier–Mellin transform is unchanged by changes in b. This property carries over to the 2D Fourier–Mellin transform as well. We conclude that the absolute value of the Fourier–Mellin transform is unchanged by expansion or contraction of the scale size of the function being transformed. This property has been exploited in pattern recognition. See, for example, Ref. [9].

Back to Contents?

14. Fourier Transforms and Digital Image Processing With *Mathematica*

While in the not-too-distant past, the most common methods for capturing and recording images were based on photographic film, today the vast majority of images are collected and processed in digital form. *Mathematica* has considerable capability to process such imagery in various ways. Digital image transformations often involve modifications to the contrast in images, geometric transformations such as removal of distortion, feature extraction, and resolution enhancement, to mention a few such operations. Here we introduce the reader to only a few of *Mathematica*'s capabilities in this broad and diverse field.

Recognizing that the Fourier plane is accessible, it is possible to place a transparency in the common focal plane that contains a desired transfer function, and the filtered image will appear in the rear focal plane of the second lens.

There is one subtlety that should be mentioned. The double-lens imaging system operates as a linear, invariant system on the complex fields. That is, the complex field in the input plane is linearly filtered to produce the complex field at the output. However, detectors in the optical region of the spectrum cannot detect complex fields. Rather, they detect intensity, which is the squared magnitude of the complex field. Therefore, what we observe at the output of the double Fourier-transforming system is the squared magnitude of the filtered field; thus there is a final nonlinear operation before observation of the result. Nonetheless, there are many examples of useful image processing operations that can be performed by such a system. We focus on one such operation in the section that follows.

15.3 Phase Contrast Imaging

In microscopy, it is often the case that specimens of interest are very weakly absorbing, making it difficult or impossible to observe them with a conventional microscope. However, those specimens typically have a refractive index profile that is significantly different than their surroundings, making them predominantly phase objects. Several different methods are available to enhance such images, namely, methods that convert phase perturbations into intensity variations observable with the properly modified microscope.

We should mention at the start that a typical conventional microscope is not unlike the optical system described above with two lenses separated by twice their common focal length. The chief difference is that the focal length of the objective lens (the first lens in the sequence) is much shorter than the focal length of the eyepiece (the second lens), leading to a magnification that is the ratio of the two focal lengths. Another difference is that the objective may not be a simple lens as we hypothesized, but rather a group of lenses, chosen to minimize aberrations and chromatic dependencies. However, the simple model with two lenses having identical focal lengths captures the essence of the system and will continue to be used here.

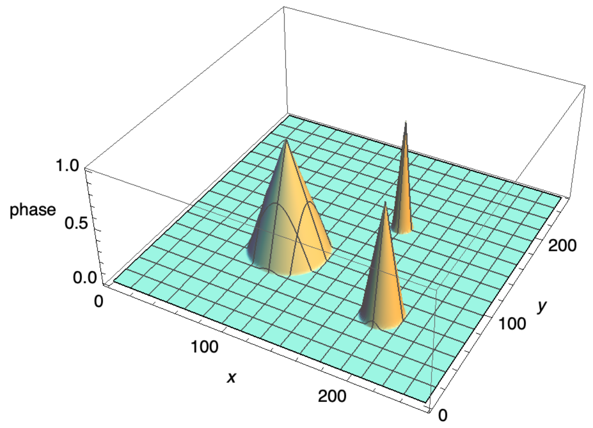
We explore some methods for making such objects visible with good contrast in what follows.

Modeling a Pure Phase Object

To test the various methods for visualizing phase objects, we must start with a simple model of such an object. As a simple model, we adopt a blank background on top of which ride three circularly symmetric phase objects of different radii and each with a triangular radial phase profile. The size of the object array is denoted by *n*, and in the interest of keeping this file as small as possible, *n* is chosen to be 256. The reader with the full *Mathematica* program can experiment with increasing the value of *n* in these examples.

In what follows we present the code to create this object and show images of the phase profile:

```
In[∘]:= n = 256;
        phaseprofile = Parallelize[Table[
             tri[Sqrt[((x - n / 8) ^2 + (y - n / 8) ^2)] / 8] + tri[Sqrt[((x + n / 8) ^2 + (y + n / 8) ^2)] / 32] +
              tri[Sqrt[((x + n / 4) ^2 + (y - n / 4) ^2)] / 16], {x, -n / 2 + 1, n / 2}, {y, -n / 2 + 1, n / 2}]];
        ListPlot3D[phaseprofile, PlotRange → All, AxesLabel → {x, y, phase}]
```

Out[∘]=


We can also convert this phase profile to an image: