

A Primer to Engineering Functions in Symbolic Terms

Victor Liu

Contents

| | |
|---|----------|
| Introduction | iv |
| Chapter 1 | |
| <i>Dusting off the tools</i> | 1 |
| 1.1 Classical functions | 1 |
| 1.1.1 Linear, polynomial, and rational functions | 1 |
| 1.1.2 Periodic functions | 1 |
| 1.1.3 Order spanning and order determining functions | 2 |
| 1.1.4 Hyperbolic and inverse trigonometric functions | 2 |
| 1.2 Special functions and the beginnings of composition | 4 |
| 1.2.1 The Gaussian | 4 |
| 1.2.2 Peaked functions | 5 |
| 1.2.3 Exponentiated inverses and maximally flat matching | 5 |
| 1.2.4 Legendre, Hermite, Bessel, Airy, and all the ones you didn't learn | 5 |
| 1.2.5 Utility (nonanalytic) functions | 6 |
| Chapter 2 | |
| <i>Composing a masterpiece</i> | 8 |
| 2.1 Operators and order twiddling | 8 |
| 2.1.1 Calculator functions | 8 |
| 2.1.2 Linear operations - Scaling, Shifting, Superimposing | 8 |
| 2.1.3 Discrete functions | 8 |
| 2.1.4 Transformations | 8 |
| Chapter 3 | |
| <i>Seeing how it's done</i> | 9 |
| 3.1 Top hat | 9 |
| 3.1.1 Direct composition | 9 |
| 3.1.2 Truncation | 10 |
| 3.2 Higher order hats | 10 |
| 3.2.1 Direct composition | 10 |
| 3.2.2 Truncation | 10 |
| 3.2.3 Self convolution | 11 |
| 3.3 Square wave and other periodic waveforms | 12 |
| 3.3.1 Square wave | 12 |

| | | |
|------------------|--|-----------|
| 3.3.2 | Sawtooth wave | 12 |
| 3.3.3 | Triangle wave | 13 |
| 3.3.4 | Arbitrary periodic waveform | 13 |
| 3.4 | Criteria based composition | 14 |
| 3.4.1 | Visually appealing band-limited square grating | 14 |
| 3.4.2 | Merit functions | 15 |
| Chapter 4 | | |
| | <i>Exercises</i> | 17 |

Introduction

This pamphlet was written to document several techniques I regularly use when constructing symbolic functional forms in order to clarify the methods. The need to express graphically described functions in symbolic form arises when computations or transformations must be performed on a function whose value is easily expressed through given characteristics and behaviors rather than in the form of an equation. Classical methods for performing this symbolic transformation exist such as Taylor series approximations, polynomial fits, or Fourier series. The problem with such methods is that these methods tend to be involved computations and are often only approximations to a given function. These interpolative methods also become problematic when exact values are required for unsampled values of the domain, which is necessary for computing Fourier transforms, for example.

The methods presented here will allow certain graphically defined functions, such as square waves or merit functions, to be quickly composed and expressed in closed form. The basis for all this is understanding the properties of basic functions that can be composed to form a complex function with the desired behavior. Examples and Mathematica code is provided to illustrate the concepts.

If you are one who skims texts, then I advise you to focus more on the examples and code of Chapter 3, since that will allow you to get started immediately by example rather than read material you probably already know.

Dusting off the tools

Let us refamiliarize ourselves with some functions that you should either already know or have heard of. The earlier portions of this chapter may be skipped if the reader is already familiar with classical functions. These functions are the basic tools that we use to compose more complicated functions.

Classical functions

Linear, polynomial, and rational functions

Your most basic function is the linear function $y = mx + b$, where m is the slope of the line and b is the y-intercept. This is not always the most useful form when performing linear interpolation, so when given a point (x_0, y_0) through which a line passes and the slope m , then one can use

$$y = y_0 + m(x - x_0) \quad (1.1)$$

Or if given two points (x_0, y_0) and (x_1, y_1) on a line,

$$y = y_0 + \frac{y_1 - y_0}{x_1 - x_0}(x - x_0) \quad (1.2)$$

The linear function is the basis by which other functions are compared when considering order, since it has a constant rate of increase.

Polynomial and rational functions are usually used to introduce functional zeros or singularities at particular locations. In particular, if a set of zeros z_i and singularities p_i are required, a function of the form

$$y = \frac{\sum(x - z_i)}{\sum(x - p_i)} \quad (1.3)$$

will satisfy the requirements. Often the function is required to be positive or negative between certain zeros or singularities, and evaluation at a test value or plotting the function will determine the sign. Each first order zero or singularity will introduce a sign change in the function, so second order zeros and singularities can be used to maintain the sign of the function.

Periodic functions

The most basic periodic functions are the trigonometric functions, in particular sin and cos. To have zero crossings at multiples of Δx , one would use an argument scaling of $y = \sin \frac{\pi x}{\Delta x}$. The other trigonometric functions have similar properties, except tan and cot have half the period of sin. Not much can be said here that cannot be found in any high school math text, so we will leave it at that.

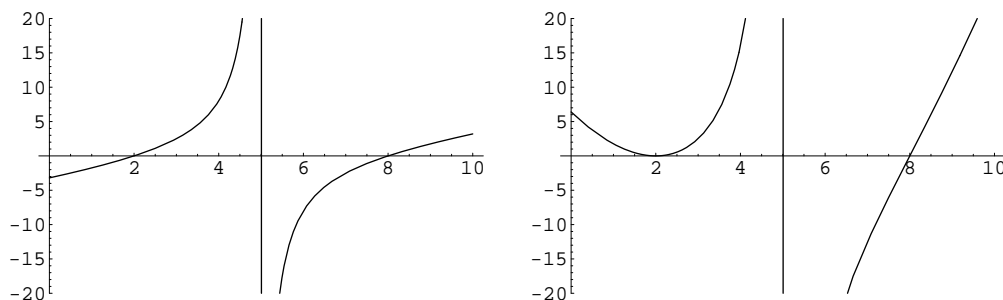


Figure 1.1: Graph of $y = \frac{(x-2)(x-8)}{x-5}$ and $y = \frac{(x-2)^2(x-8)}{x-5}$. Note the second order zero preserves sign on either side of the zero.

Order spanning and order determining functions

The exponential function is one that grows by a constant multiplicative factor for each unit increment on the domain. Thus it is used to span orders of magnitude; a linear argument is transformed to a set of orders of magnitude. The logarithm, being the inverse function, is order determining, meaning it will return the number of orders of magnitude of a given value. In particular, the base 10 logarithm can be used to find the number of nonfractional digits in a number or in general the location of the decimal point.

When testing the behavior of complicated functions, providing an exponential as an input is a quick way to determine behavior at very large and very small input values. Note of course, that it nonnegative, so it does not span all possible input values. Similarly the logarithm can be used to understand the growth of functions when provided a rapidly increasing function as an argument. It is also useful for order comparison, since exponentiation is transformed to multiplication. The laws of exponents and logarithms are essential when working with these functions; consult any calculus textbook for these.

Hyperbolic and inverse trigonometric functions

The hyperbolic functions have a reputation for being arcane and is often not well understood. They are simply compositions of exponential functions, and this is graphically obvious as the asymptotes are either growing or decaying exponentials. A particularly useful function is the hyperbolic tangent, plotted below. It can be used when a ramping function is required. In general, if a ramp is required between two values $y = y_0$ and $y = y_1$ with corners at $x = x_0$ and $x = x_1$, we would use

$$y = y_0 + \frac{y_1 - y_0}{2} \left(\tanh \left[\frac{2}{x_1 - x_0} \left(x - \frac{x_0 + x_1}{2} \right) \right] + 1 \right) \quad (1.4)$$

The asymptotic function is (using Heavisides),

$$y = y_0 + \frac{y_1 - y_0}{x_1 - x_0} (x - x_0)H(x - x_0) - \frac{y_1 - y_0}{x_1 - x_0} (x - x_1)H(x - x_1) \quad (1.5)$$

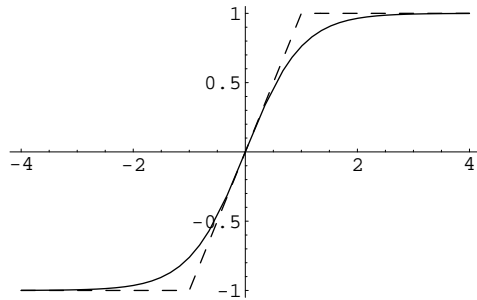


Figure 1.2: Graph of $y = \tanh x$ and its asymptotes in dashed lines.

The general form given other parameters is left as an exercise for the reader. The hyperbolic tangent equation is preferred over the asymptotic expression because it is smooth and everywhere differentiable, thus making it suitable for use in merit functions for optimization.

A function with behavior similar to the hyperbolic tangent is the inverse tangent function, shown below with its asymptotes. In general, the arctangent approaches its asymptotes

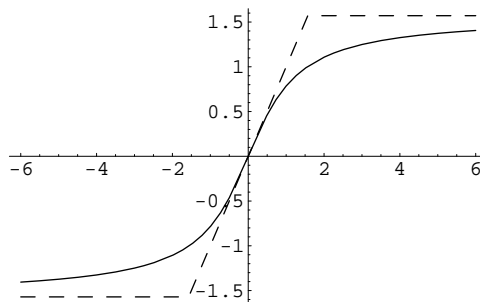


Figure 1.3: Graph of $y = \tan^{-1} x$ and its asymptotes in dashed lines.

more slowly is therefore better suited to applications that require a more spread out ramping behavior. Given the same ramp parameters as above, the generic form is

$$y = y_0 + \frac{y_1 - y_0}{\pi} \left(\tan^{-1} \left[\frac{\pi}{x_1 - x_0} \left(x - \frac{x_0 + x_1}{2} \right) \right] + \frac{\pi}{2} \right) \quad (1.6)$$

with the same asymptote form as above.

Another inverse trigonometric function that is useful is the inverse sine (the inverse cosine can be easily transformed into the inverse sine), shown below. It is approximately linear with vertical asymptotes at each end of the domain. Instead of interpreting the nonlinearity at the ends as asymptotes, it can be thought of as superlinear behavior, in which the function increases with order greater than linear, while the asymptotic behavior is usually useless. The superlinear behavior can be used to model deviation from linearity. The deviation (both absolute and percentage) between the asymptote and the actual value at the ends of the domain is left as an exercise for the reader. Given this, imagine how one might use the sine function to model sublinear behavior.

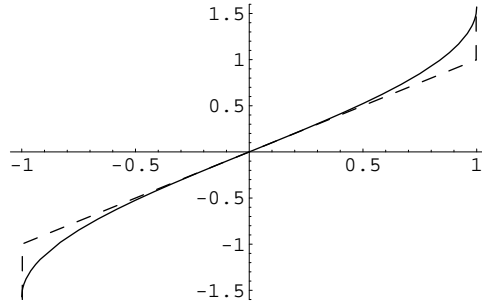


Figure 1.4: Graph of $y = \sin^{-1} x$ and its asymptotes in dashed lines.

Special functions and the beginnings of composition

The Gaussian

The Gaussian is the de facto peak function, with functional form invariant under Fourier transformation. For a unity height peak with FWHM Δx located at $x = x_0$, the general form is

$$y = \exp \left[-\frac{(x - x_0)^2}{2(\Delta x/2)^2} \right] \quad (1.7)$$

Being infinitely smooth, it is preferable over composed piecewise linear peak functions if computational complexity of the expression is not a concern. The derivatives of the Gaussian have successively more peaks, with odd numbers of derivatives producing odd functions, and even numbers of derivatives producing even functions. Note also that the peaks of the next derivative occur at the inflection points of the current derivative and grow in height.

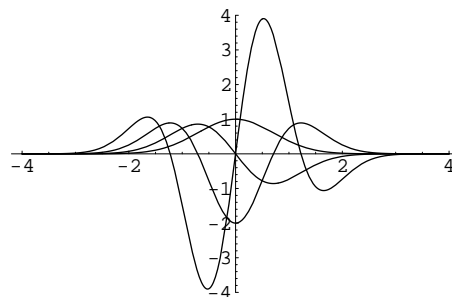


Figure 1.5: Graph of a Gaussian and its first three derivatives.

Peaked functions

There are many other functions with peaks. Perhaps the simplest is the rational function

$$y = \frac{1}{|x|^n + 1} \quad (1.8)$$

which has the property that the decay rate is polynomial in nature (as x^{-n}). The absolute value is necessary for noninteger n . Also, the hump is only smooth when n is an integer. For a slow decay, n can be an arbitrarily small positive number (or variable for that matter).

Exponentially decaying one sided peaks such as xe^{-x} , x^2e^{-x} , etc. are useful since the growth rate can be easily specified independently of the decay rate. The exponent can also contain powers of x for faster decay.

This is merely a sampling of simple peaked functions. More complicated ones exist but seldom is such complexity necessary.

Exponentiated inverses and maximally flat matching

Once during a physics lecture a question was posed: "what is the simplest increasing function that has every derivative zero at the origin?"¹ One can guess the solution by requiring $\frac{d^n}{dx^n}y(0) = 0$ and $\frac{dy}{dx} > 0$ and finding

$$y = e^{-\frac{1}{x}}$$

The function is maximally flat at the origin and can be used when a smooth transition between curves of different slopes are required. The limiting value at $x = \infty$ is unity, so there must be an inflection point in the curve. Joining piecewise parabolic sections together would achieve the same goal, but with the problem that not every derivative is smooth, as is true in this case. Different powers of x may also be used to change the rate of increase near the origin, but only powers greater than 1 demonstrate the zero-derivatives property. When given the order n of x and the desired location of the inflection point (x_i, y_i) , the general form is

$$y = y_i \exp \left[-\frac{n+1}{n} \left(1 - \frac{x_i}{x}\right)^n \right] \quad (1.9)$$

Note that the requirement for a fixed y_i change the function's limiting value to $y(\infty) = y_i \exp \left[\frac{n+1}{n} \right]$

Legendre, Hermite, Bessel, Airy, and all the ones you didn't learn

Additional special functions that have interesting properties are the Legendre, Hermite, Chebyshev, etc. basis polynomials used in eigenfunction expansions. There are also Bessel,

¹This was during a thermodynamics course while solving for the dependence of temperature on entropy.

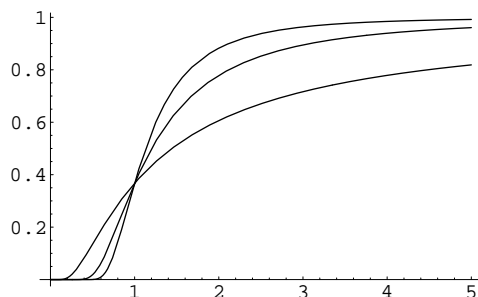


Figure 1.6: Graph of a exponentiated inverses $e^{-x^{-1}}$, $e^{-x^{-2}}$, and $e^{-x^{-3}}$. These are listed in the rising order in the graph.

Airy, Gamma, etc. functions defined by differential equations that have very useful properties. These are generally either unnecessary because a given functional behavior can be described with more elementary functions or overly sophisticated when simple expressions suffice to satisfy a given set of characteristics. There exist handbooks of special functions that detail their properties, to which the reader is directed for further inquiry.

Utility (nonanalytic) functions

Surprising, the nonstandard, nonsmooth functions described here are the most useful when composing functions.

Let us begin with the Dirac delta function, which can be expressed as the limit of a normalized Gaussian as the FWHM goes to zero, while maintaining the area under the curve normalized, creating an infinite spike. Formally,

$$\int_{-\infty}^{\infty} f(x)\delta(x - x_0)dx = f(x_0) \quad \int_{-\infty}^{\infty} \delta(x)dx = 1 \quad (1.10)$$

Of course, there is no need to integrate to infinity, since the function is mostly zero. Taking integrals immediately around x_0 in the first case would have sufficed.

Notice now that we can define the Heaviside step function as the integral of the delta function:

$$H(x) \approx \int_{-\infty}^x \delta(t)dt \quad (1.11)$$

This is only approximately true because the behavior at $x = 0$ is unclear. A better definition is

$$H(x) = \begin{cases} 1 & x > 0 \\ \frac{1}{2} & x = 0 \\ 0 & x < 0 \end{cases} \quad (1.12)$$

Integrating this again we get the ramp function

$$R(x) = \int_{-\infty}^x H(t)dt = \frac{1}{2}(x + |x|) = \begin{cases} x & x > 0 \\ 0 & x \leq 0 \end{cases} \quad (1.13)$$

The sign function, which returns the sign of a value, is related to the Heaviside by

$$\text{Sign}(x) = 2H(x) - 1 = \begin{cases} 1 & x > 0 \\ 0 & x = 0 \\ -1 & x < 0 \end{cases} \quad (1.14)$$

The traditional absolute value function can now be expressed in terms of the sign function:

$$|x| = x \text{Sign}(x) = \begin{cases} x & x > 0 \\ -x & x \leq 0 \end{cases} \quad (1.15)$$

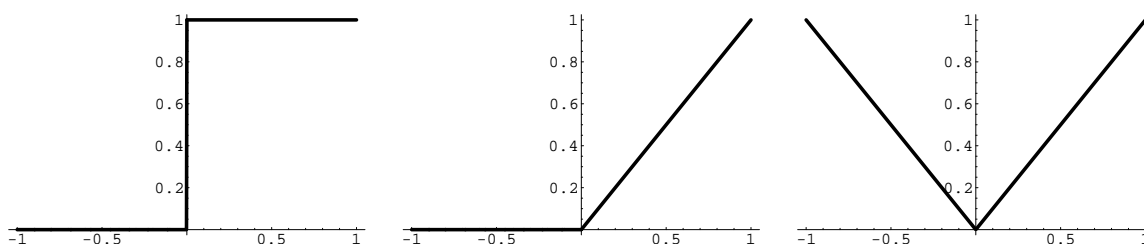


Figure 1.7: In order, the Heaviside, ramp, and absolute value functions.

It is not the definition of these functions that is useful, but its graphical properties. When multiplied against a function, the Heaviside function picks out a range of values and zeros the rest. Additionally, the corners and the ability to scale functions on a certain part of a domain are incredibly useful, as we shall see.

Several others worth of mention are the floor and ceiling function which round a value to the next lowest or highest integer. Note that $\text{Floor}(x) = \text{Ceiling}(x) - 1$ and traditional rounding is then $y_{\text{round}}(x) = \text{Floor}(x + \frac{1}{2})$. The plots are shown below. These will be useful when generating periodic waveforms.

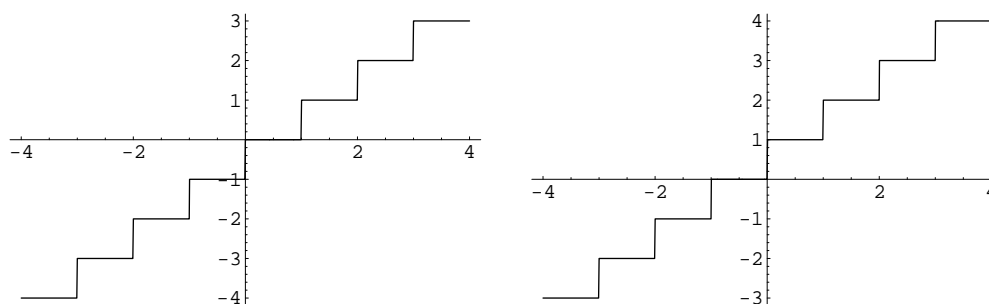


Figure 1.8: The floor and ceiling functions.

Composing a masterpiece

This is a short chapter about operators and what can be done with them.

Operators and order twiddling

Calculator functions

The most basic operators are addition and the like, which everyone should know. Their functions when dealing with the order of growth of functions is not always clear to everyone. It is important to keep in mind the laws of exponents; multiplying adds orders of magnitude (exponents add), and exponentiation multiplies orders of magnitude (exponents multiply). Keep this in mind when spanning domains with a range of orders of magnitude is necessary.

Linear operations - Scaling, Shifting, Superimposing

Scaling functions vertically is equivalent to multiplying by a constant factor: $Ay(x)$ is a function A times taller than $y(x)$. To introduce a horizontal shift, we know that $y(x - x_0)$ is the function $y(x)$ shifted x_0 units to the right. And superimposing functions simply means adding them together. Hopefully nothing new has been introduced here.

Discrete functions

Sometimes we are interested in expressing functions on a discrete domain (such as the set of integers) in closed form. One simply has to remember that the functional behavior between the discrete points is completely irrelevant. For example, $\sin(2\pi x)$ is a constant function (equal to 1) when evaluated over the integers. In this case, exact interpolations are sufficient to express the underlying function.

Transformations

An intimate understanding of the properties of the Fourier or Laplace transforms can go a long way in deriving more complicated functions from simpler ones very quickly. The convolution (usually with delta functions) is an incredibly useful tool, and will be apparent later. Once again, many differential equations and signal processing texts cover this topic in incredible detail, so no further discussion is provided here.

Seeing how it's done

If you read any chapter, read this one. The examples here are far more instructive than anything described previously.

Top hat

It is often necessary to create a function to pick out values of another function on a certain set of the domain. What one needs is a boxcar function; one that is unity over the region of interest and zero elsewhere.

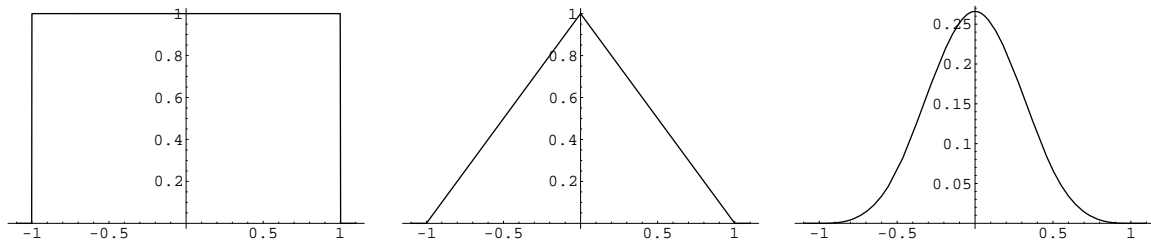


Figure 3.1: The top hat (boxcar), witch's hat, and parabolic hat. All functions have been scaled to be nonzero on $(-1, 1)$, particularly the parabolic hat which is derived from convolution.

Direct composition

The top hat has two steps, one going up at $x = -1$ and one going down at $x = 1$. We see that two superimposed Heavisides with steps at the appropriate locations will do the trick:

$$\text{hat}_{\text{top}}(x) = H(x + 1) - H(x - 1) \quad (3.1)$$

In fact, the plot above was generated using this Mathematica code:

```
Plot[UnitStep[x + 1] - UnitStep[x - 1], {x, -1.1, 1.1}]
```

This is the simplest way of creating the function in a sensible manner. One could also use

$$\text{hat}_{\text{top}}(x) = \frac{1}{2} [\text{Sign}(-(x + 1)(x - 1)) + 1] \quad (3.2)$$

where the polynomial within the sign function has zero crossings at $x = \pm 1$ with sign adjusted accordingly. The corresponding Mathematica definition is

```
TopHat[x_] := (Sign[(-(x + 1)(x - 1)) + 1] / 2;
```

Truncation

One could also think of the top hat as a truncated version of the constant function $y = 1$. We know that multiplication by the Heaviside will essential truncate the function to a semi-infinite domain, so with one shifted and one shifted and reversed Heaviside, we can obtain the same result:

$$\text{hat}_{\text{top}}(x) = H(x + 1)H(1 - x) \quad (3.3)$$

```
TopHat[x_] := UnitStep[x + 1] UnitStep[1 - x];
```

Higher order hats

Higher order hats that have smoother characteristics are also sometimes necessary, and the first two are shown in Figure 3.1. Let us see how we may compose these functions.

Direct composition

Once again in directly forming the graph, we notice three corners in the graph, one going up, one peak, and one leveling off. This indicates that three ramp functions are necessary. Each one must have slope to account for the change in slope necessary in going from each piece to the next. The first segment on $[-1, 0]$ requires slope 1, then the slope changes to -1 , and then back to 0. This is created with

$$\text{hat}_{\text{witch}}(x) = (x + 1)H(x + 1) - 2xH(x) + (x - 1)H(x - 1) \quad (3.4)$$

The code used to generate the previous plot is

```
Plot[(x + 1)UnitStep[x + 1]
      - 2x UnitStep[x]
      + (x - 1)UnitStep[x - 1], {x, -1.1, 1.1}]
```

Truncation

We notice the graph resembles a truncated upside-down absolute value function, so we use just that expression:

$$\text{hat}_{\text{witch}}(x) = H(x + 1)H(1 - x)(1 - |x|) \quad (3.5)$$

```
a[x_] := UnitStep[1 + x]UnitStep[1 - x](1 - Abs[x])
```

Self convolution

This is by far the most direct method for computing the witch's hat function, since we notice from the graph that it is simply the convolution of the top hat with itself (with additional scaling factors in width and amplitude), which means the Fourier transform gets squared. Of course, performing this computation by hand is tedious, but most often, the reasons for composing such functions is so that a computer program can perform symbolic manipulations on the expression. Embarrassingly enough, Mathematica cannot perform the inverse transformation required for the witch's hat:

```
TopHat[x_] := UnitStep[x + 1] - UnitStep[x - 1];
WitchHat[x_] := InverseFourierTransform[
    FourierTransform[TopHat[X], X, k]^2, k, X
] /. X -> x;
(***** Fails to find the inverse transform *****)
```

Changing the definition of the top hat function to transform, however, allows us to evaluate the convolution:

```
TopHat[x_] := UnitStep[x + 1]UnitStep[1 - x];
WitchHat[x_] := InverseFourierTransform[
    FourierTransform[TopHat[X], X, k]^2, k, X
] /. {X -> 2x};
WitchHat[x]
Plot[%, {x, -1.1, 1.1}]
```

This function is a vertically scaled version of the one in Figure 3.1 since the value at $x = 0$ is $\sqrt{2/\pi}$ but the horizontal scaling was corrected with the substitution of $2x$. The code to generate the parabolic hat, the convolution of the witch's hat with itself, can be performed with

```
WitchHat[x_] := (x + 1)UnitStep[x + 1]
    - 2x UnitStep[x]
    + (x - 1) UnitStep[x - 1];
ParabolaHat[u_] := InverseFourierTransform[
    FourierTransform[WitchHat[x], x, k]^2, k, x
] /. {x -> 2u};
ParabolaHat[x]
Plot[%, {x, -1.1, 1.1}]
```

This code produces the graph in Figure 3.1.

This procedure of using convolution typically works when one of the functions is very simple, such as delta functions, sinusoids, and the like. Being a hit-or-miss method, one must simply attempt the computation in Mathematica.

Square wave and other periodic waveforms

Square wave

The need to generate a square wave is quite common. A very rudimentary and obvious form is

$$y_{\text{square}}(x) = \text{Sign}(\sin \pi x) \quad (3.6)$$

and the plot is shown below. This can be interpreted in two ways. The first is that we find

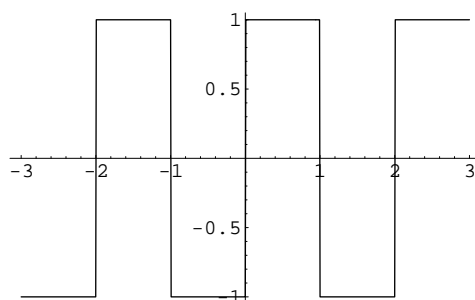


Figure 3.2: An example square wave.

the Sign of the value of sin, which is perhaps the more intuitive way of thinking about the expression. The second is that the sin function wraps the domain of the Sign function onto itself, so that we periodically evaluate the Sign function on the domain $[-1, 1]$. It is this second method that is often forgotten but extremely useful, which we will demonstrate for the sawtooth wave later.

To compute a square wave with a period T and a duty cycle of p where p is the fraction of time the wave is high between 0 and 1, we have the general formula

$$\text{Sign} \left[\cos \left(\frac{2\pi x}{T} \right) + \sin \pi \left(p - \frac{1}{2} \right) \right] \quad (3.7)$$

Sawtooth wave

Using the idea of domain wrapping and the sine function as a foundation again, we can develop a sawtooth wave. The sine function rises sublinearly over half its period and then decreases sublinearly over half its period. We need to use the output of the sine function as a domain wrapped input to a function that takes a sublinear input and makes it linear. The exact function to perform the undoing is the inverse sine function. So we have a very simple expression

$$y_{\text{sawtooth}}(x) = \frac{2}{\pi} \sin^{-1} \sin(\pi x) \quad (3.8)$$

which is graphed below.

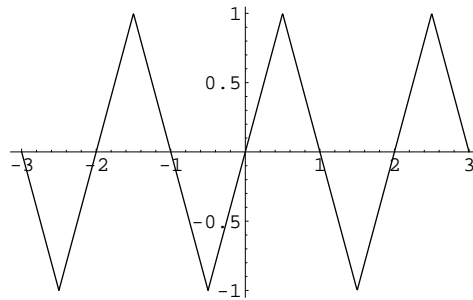


Figure 3.3: A simple sawtooth wave.

Triangle wave

There are several approaches that we may consider. First, the triangle wave is a series of truncated ramp functions, so we may convolute the truncated ramp function with a properly spaced delta function comb. However this method is difficult to express in Mathematica, and is perhaps an easier way to compute by hand. A second way is to consider the floor function, which is piecewise constant with a general linear increase. The error between the floor function and the linear function is a triangle wave:

$$y_{\text{triangle}}(x) = x - \text{Floor}(x) \quad (3.9)$$

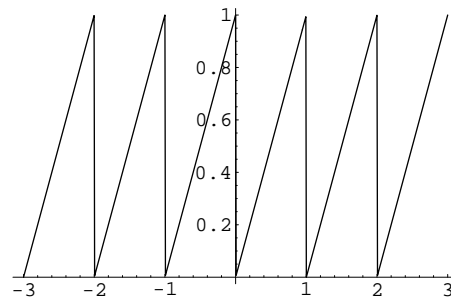


Figure 3.4: A triangle wave.

Arbitrary periodic waveform

Now we are ready to present a general method for generating periodic waveforms. Suppose a waveform described by $u(x)$ on $[0, 1)$ is one period of the desired waveform. Using the idea of domain wrapping, we simply need to periodically evaluate $u(x)$ with a linear input, and the triangle wave does just that. So then given a unit of the waveform $u(x)$, we can express the periodic waveform $U(x)$ as

$$U(x) = u(x - \text{Floor}(x)) \quad (3.10)$$

An example for

$$u(x) = x^2 - \frac{3}{2} \left(x - \frac{1}{3}\right) H\left(x - \frac{1}{3}\right)$$

is given below.

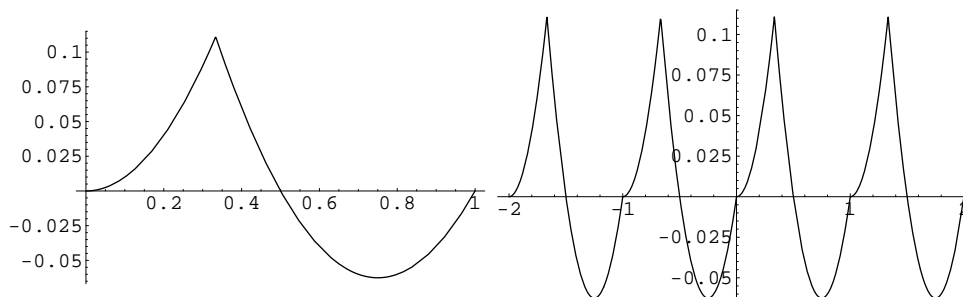


Figure 3.5: The unit to be repeated $u(x)$ is shown on the left, and the periodic waveform is on the right.

Criteria based composition

Sometimes it is not known exactly how a function is defined, but we know the approximate properties the function should have. The problem then becomes to find a simple expression satisfying the function design criteria. Although traditional methods such as Taylor series expansions or polynomial fitting can produce the desired results, intuition can often produce even simpler results that capture the essence of the problem. Other times, it is impossible to satisfy constraints without simply observing the resulting function. Here are several examples to illustrate these points.

Visually appealing band-limited square grating

This problem has come up in an image filter design study where a band-pass filter for certain square wave patterns (gratings) must be found. The distribution of frequencies must be “visually appealing.” The grating must essentially be a frequency varying square wave, but the variations must appear evenly distributed, so the problem is essentially one of describing the frequency variation.

Let us pose several criteria first. The low frequency is $f_L = 1$ and the fast period is $f_H = 10$, and the constants are defined by

```
Grating[x_] := Sign[Sin[\[Pi] x]];
fL = 1;
fH = 10;
d = 10;
```

Consider first a linear ramping of the frequency from a low value to a high value. The result is shown in the first graph of Figure 3.4.1. The code to produce is

```
F[x_] = x (fL + (fH - fL) x/d);
Plot[Grating[F[x]], {x, 0, d}, PlotPoints -> 10000]
```

The result is very dense at high frequencies and long periods are under represented.

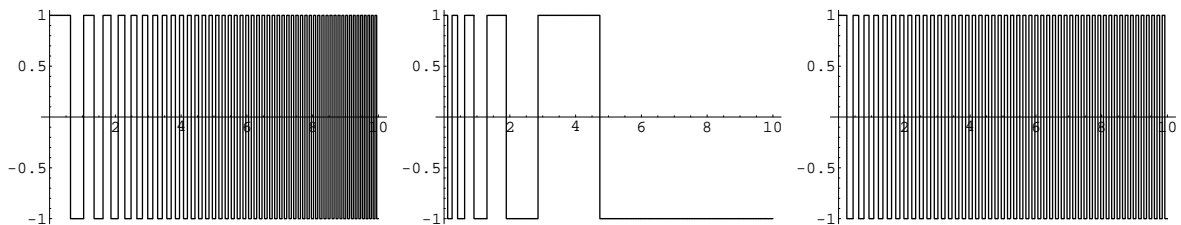
Taking instead the period to be linearly varying gives the result in the middle graph of Figure 3.4.1 and code

```
TL = 1;
TH = 0.1;
F[x_] = x/(TH + (TL - TH)x/d);
Plot[Grating[F[x]], {x, 0, d}, PlotPoints -> 10000]
```

The opposite problem occurs, and the result is much worse.

Now we can try choosing a polynomial variation in frequency and tweaking the exponent. After testing several values, the final result is shown in the last graph in Figure 3.4.1 and the code to produce it is

```
F[x_] = x(L + (H - L)(x/d)^0.3);
Plot[Grating[F[x]], {x, 0, d}, PlotPoints -> 10000]
```



Merit functions

In the processes of optimizing systems, there are often certain goals and constraints that cannot be easily satisfied when a direct figure of merit is supplied for optimization. For example, suppose a system has multiple parameters that can be varied, and multiple figures of merit that must be maximized or tuned to a specific value. Naive optimization of all figures of merit simultaneously usually leads to cases where only a few or none of the figures of merit are acceptable. Therefore some amount of hand tuning of the function to optimize must be made. This mostly involves the design of a function that is maximal when figures of merit are maximal, but does not allow any one of them to be significantly worse than the others.

An example for the simultaneous maximization of several figures ($f_i(\bar{x})$) at once can be done by maximizing the product of various powers of each figure:

$$M(\bar{x}) = \prod_i [f_i(\bar{x})]^{n_i} \quad (3.11)$$

If the values in this expression can be large, then the product may cause an overflow, so we may simply maximize the log:

$$M(\bar{x}) = \prod_i [n_i \log f_i(\bar{x})] \quad (3.12)$$

Another problem arises when the f_i may be zero but there are usually ways to design the f_i to be strictly positive functions while still capturing the essence of the figure. This all works decently well assuming the n_i can be set independent of the figures, which is usually not the case. In many cases, a peaked function must be used to coerce the merit figures toward a certain value, and iterative optimization with updated initial guesses can usually be successful.

Exercises

1. Create a periodically parabolic (increasing from zero slope at each period) wave. Do this with and without domain wrapping.
2. Suppose we have a function

$$y(x) = \begin{cases} -\frac{1}{2}x - 3 & x < -1 \\ 2x + 1 & x > 1 \end{cases} \quad (4.1)$$

Supply a definition for $y(x)$ on $[-1, 1]$ that makes $y(x)$ everywhere differentiable using the fewest number of quadratic curves. With the supplied piece, how many times can $y(x)$ be differentiated at $x = \pm 1$? What is the jump in the last possible derivative about $x = \pm 1$?

3. Repeat the above problem except now use a pair of exponentiated inverses to patch $y(x)$. How many derivatives can be taken at $x = \pm 1$? How many at the joining of the two exponentials? Explore the effect of the order n on the number of derivatives that exist.
4. Prove that there exists a domain scaling of the hyperbolic tangent on the positive domain that yields another linearly translated hyperbolic tangent as a result. (Hint: consider the definition of \tanh .)