Fourier Transform Techniques

by Michael Prior-Jones

based on lectures and course material by G.J.Ritchie and A.I.Tew

Recommended Textbook:

Soliman & Shrinath- Continuous and Discrete Signals and Systems, ISBN 0135184738

Brief introduction to the three forms of transform analysis:

- Phasor analysis – used for sinusoidal, periodic waveforms
- Fourier series – used for periodic but non-sinusoidal waveforms
- Laplace and Fourier transforms – used for non-periodic and non-sinusoidal waveforms

1 Fourier Series

Up until now, we've always used phasor analysis to determine a circuit's response to a sinusoidal signal. Whilst phasor analysis is relatively straightforward, it's limited strictly to sinusoidal waves- it simply won't work any other way. What we're going to look at now is the behaviour of other periodic functions- square waves, sawtooth waves, triangle waves and so on.

Fourier (pronounced foo-ree-ay), who was a nineteenth-century mathematician, realised that any practical periodic function can be represented as the sum of sinusoids. These sinusoids are called the components of the function.

The principle of superposition allows us to find the response of a circuit using phasor analysis for each frequency component.

1.1 Trigonometric Fourier Series

For a periodic function (one that repeats every T seconds), Fourier's theorem states that:

> "Any practical periodic function can be represented as an infinite sum of sin and cos functions that have frequencies which are integer multiples of the fundamental frequency."

Got that ? Any waveform that repeats itself can be described as the sum of an infinite series of sine and cosine functions. Useful!

The series is defined as follows:

$$f(t) = a_0 + \sum_{n=1}^{\infty} \left(a_n \cos n\omega_0 t + b_n \sin n\omega_0 t \right)$$

a_0 is the dc coefficient
a_n and b_n are the Fourier coefficients (there are n of each)
ω_0 is the fundamental angular frequency.

Fourier analysis involves finding the Fourier coefficients.

Some useful integral relationships:

Because sinusoidal functions are highly symmetrical, integrating them over an integer number of periods (period = T) tends to produce very simple answers.

$$\int_0^T \sin(nwt)\sin(mwt)dt = 0 \text{ except where } n = m, \text{ where the answer is } \frac{T}{2}$$

The same is also true for two cos functions.

$$\int_0^T \sin(nwt)\cos(mwt)dt = 0$$

The functions which produce a zero result are said to be orthogonal.

Fourier series analysis

This is a simple step-by-step method, assuming that you can do the integrals…

Step 1- find the dc coefficient, a_0. This is simply the average of the waveform:

$$a_0 = \frac{1}{T}\int_0^T f(t)dt$$

Step 2- find the cosine coefficients, a_n

$$a_n = \frac{2}{T}\int_0^T f(t)\cos n\omega_0 t dt$$

Step 3- find the sine coefficients, b_n

$$b_n = \frac{2}{T}\int_0^T f(t)\sin n\omega_0 t dt$$

This gives the coefficients in their trigonometric form- to convert to the more useful amplitude and phase form, use the following:

$A_n = \sqrt{a_n^{\,2} + b_n^{\,2}}$ amplitudes of harmonics

$\phi_n = -\tan^{-1}\left(\frac{b_n}{a_n}\right)$ phases of harmonics

Plotting A_n against frequency gives an amplitude spectrum.
Plotting ϕ_n against frequency gives a phase spectrum.
Together they form a frequency spectrum.

Gibbs' Phenomenon

As you sum the Fourier series, you get a closer and closer approximation to the waveform. However, in areas close to discontinuities ("sharp edges" on square waveforms) you get an overshoot and a damped oscillation (or "ringing"). The height of the overshoot is always 9% of the waveform amplitude, regardless of the number of terms you sum.

1.2 Symmetry considerations

Many periodic waveforms also exhibit some form of symmetry. By spotting the symmetry in a waveform, you can often drastically reduce the amount of calculations you need to do.

Odd symmetric waveforms (you might remember from A-level maths that an odd function is one where $f(-t) = -f(t)$ and has rotational symmetry about the origin) generate sine terms only ($a_n=0$). Odd waveforms never have a dc coefficient ($a_0=0$).

Likewise, **even symmetric waveforms** (where $f(-t) = f(t)$ and there is thus mirror symmetry about the y-axis) generate cosine terms only ($b_n=0$)

Half-wave symmetric waveforms (where $f\left(t \pm \frac{T}{2}\right) = -f(t)$ and each half-cycle is the mirror image in the x-axis of the previous) are rather more complex. The a_n and b_n terms, where n is an even number, are 0.

1.3 Circuit applications of the Fourier Series

If you're very sad, you might realise that Fourier Series allows you to carry out a phasor analysis of a circuit driven non-sinusoidal periodic waveform. You simply replace the waveform's source with a number of sources wired together (series for voltage sources, parallel for current sources), where each source represents one term of the Fourier Series for the original waveform. Then, by the magic of superposition, you can analyse the effect of each source in turn, and when you've done this, you can add all the answers together. This is very laborious. Get a computer to do it, or use a Laplace Transform…

1.4 Exponential form of the Fourier Series

It can often be much more straightforward to transform the sines and cosines of the trignometric series into the form of complex exponentials.

This simplifies down to:

$$f(t) = \sum_{n=-\infty}^{n=\infty} c_n e^{jnw_0 t}$$

Where the coefficients c_n are defined as:

$$c_n = \frac{1}{T}\int_0^T f(t)e^{-jnw_0 t}\,dt$$

Of course, there are relationships between the two forms:

$$a_n - jb_n = 2c_n$$

$$|c_n| = \sqrt{\frac{a_n^{\,2} + b_n^{\,2}}{2}}$$

$$\angle c_n = -\tan^{-1}\left(\frac{b_n}{a_n}\right)$$

The modulus of c_n is the amplitude spectrum.

In the exponential form, you use the concepts of equal and opposite positive and negative frequencies. This may seem a little weird, but you get used to it…

The sinc function:

This is a handy shorthand for a common waveform- the spectrum of a square pulse:

$$\text{sinc}(x) = \frac{\sin x}{x}$$

and sinc(0)=1 (by L'Hopital's Rule)

1.5 Parseval's Theorem

This important theorem concerns the power of the components of a signal. It states that the power of a signal is equal to the sum of the powers of the individual sine and cosine components. Thus, removing one component (with a filter) reduces the overall signal power by the power of that component.

Basically, this lets us find the power spectrum (dead handy for RF work!) easily:

$$P = \sum_{n=-\infty}^{\infty} |c_n|^2$$

2 The Fourier Transform

A Fourier series describes the spectrum of a periodic function as an infinite series of harmonics.

A Fourier transform describes the spectrum of a non-periodic function (in fact, almost any function you like) as an integral. The use of the Fourier transform allows you to go between the time domain and frequency domain, regardless of the waveform.
You can derive the transform from considering the series describing the spectrum of a train of pulses, where the pulse width remains constant, but the time between pulses is allowed to tend to infinity. This is, however, rather tedious, and I'm not going to describe it further here!

2.1 Definition of the Fourier Transform

The upshot of it all is that:

$$F(w) = \int_{-\infty}^{\infty} f(t)e^{-jwt}$$

This is the Fourier transform.
You might notice that it's a special case of the Laplace Transform, when s=jω
It's inverse is also rather handy, and is thus:

$$f(t) = \frac{1}{2p}\int_{-\infty}^{\infty} F(w)e^{jwt}$$

Things to bear in mind when doing Fourier transforms:
- your integrals must converge. That is, the area enclosed between the waveform and the time axis must be finite. The infinite ramp waveform does not have a Fourier transform.

2.2 Properties of the Fourier Transform

- It's linear- that is, multiplying your function by a constant in the time domain results in the function in the frequency domain being multiplied by the same constant. Mathematically:

Time	Frequency
$af_1(t) + bf_2(t)$	$aF_1(w) + bF_2(w)$

- It handles timeshifts. Shifting your waveform in terms of time produces a change of phase in the frequency domain The maths of this are:

Time	Frequency
$f(t - t_0)$	$F(w)e^{-jwt_0}$

- It describes the process of modulation. Mixing a waveform up or down by a fixed frequency produces a modulation equation in the time domain:

Time	Frequency
$F(w - w_0)$	$f(t)e^{jw_0 t}$

- You can carry out a time axis scaling:

Time	Frequency
$f(at)$	$\frac{1}{\lvert a \rvert} F\left(\frac{w}{a}\right)$

- Axis reversal: The negative frequency spectrum is exactly symmetrical with the positive frequency spectrum.

Time	Frequency
f(t)	$F(w) = F(-w)$

- Duality- this is the really clever bit! Basically, any function f(t) has a Fourier transform F[f(t)] which we'll call g(ω). The clever bit is, that if you take g(ω) and substitute in to give g(t) and then take the transform again: F[g(t)] gives f(ω). So a square pulse in terms of time gives a sinc function in terms of frequency. But taking the sinc function in terms of time gives a square pulse-like frequency response.
- Integration and Differentiation: integrals and differentials in the time domain become multiplications and divisions in the frequency domain:

Time	Frequency
$\frac{df(t)}{dt}$	$jwF(w)$
$\int f(t)dt$	$\frac{F(w)}{jw}$

2.3 Applications of the Fourier Transform

- Circuit analysis: it is possible to carry out circuit analysis using a Fourier Transform, but it's quite frankly not worth the effort. The Laplace Transform is a much more suitable tool for this.
- Frequency responses and spectra: you can (obviously) use a Fourier transform to tell you the frequency response of a circuit, or the spectrum of a signal. Modern spectrum analysers use a Fast Fourier Transform (FFT for short) circuit to display spectra in real time.

The Fourier transform is most commonly found in communications systems. The ability to examine the spectrum of a signal is immensely useful, and the Fourier transform has also found its way into other systems, such as demodulating digital radio and TV signals.

2.4 Fourier vs Laplace

Fourier Transforms and Laplace Transforms both have their strengths and weaknesses. Laplace Transforms are more general, and much more easily solved algebraically. But they can only deal with cases where time >=0, and thus are most frequently used for transient behaviour analysis of circuits.
Fourier Transforms are quite difficult to solve by hand, but modern digital techniques can evaluate accurate approximations to them very rapidly. Because the Fourier Transform gives a frequency-domain output (as against Laplace's weird s-plane), they're most commonly used to find the spectrum of a signal. They can, of course, deal with all values of time.

3 Linear Time Invariant systems and Convolution

A linear, time-invariant system (or LTI system) has (obviously!) the following properties:

- it's linear
- it's time-invariant.

After that stunning piece of stating the obvious:

3.1 Definition of linearity

A linear system has the following properties:

- superposition applies: $x_1(t)+x_2(t)=y_1(t)+y_2(t)$
- it's homogenous: if $x_1(t)$ corresponds to $y_1(t)$, then $ax_1(t)$ corresponds to $ay_1(t)$.

You can test a system for linearity by taking two points (x_1,y_1) and (x_2,y_2) and then: $ax_1+bx_2=ay_1+by_2$ if the system is linear. a and b are arbitrary constants.

3.2 Definition of time-invariance

A time-invariant system has a constant time shift: if we put in a function with a timeshift, we get the same timeshift on the output function.

3.3 Properties of LTI systems

If you take an LTI system, and subject it to a unit impulse (in the time domain), then it will output its impulse response h(t), which gives a measure of how the system behaves. However, the Fourier transform of the unit impulse is a flat spectrum of infinite bandwidth (white noise) and the output of the system will be (in frequency terms) the frequency response H(t). Both impulse and frequency responses are useful. It's possible, thus, to find impulse response without using a unit impulse, simply by taking the inverse Fourier transform of the frequency response. This can be jolly useful, as going down to stores for a unit impulse generator isn't always practical- remember that a unit impulse has infinite amplitude and takes zero time to occur, but has a unit area!

3.4 Applications of impulse response

Impulse responses are less common than frequency responses, but they do have their uses. They're commonly used in acoustics, where you're not usually interested in bandwidths but reverberation times. Imagine that you're at a service in York Minster, and that someone drops something, which makes a sudden noise. This echoes around the building for several seconds. Using an impulse response plot of a building, you can easily determine how long an echo will take to die away. Concert halls (such as the Royal Festival Hall in London, and famously Symphony Hall in Birmingham) are designed to have relatively short reverberation times, so that the harmonies aren't smeared out by the reverberation of the building.

3.5 Convolution

You might be interested to know what happens when you multiply in the frequency domain. On second thoughts, perhaps you wouldn't, but I'm going to describe it anyway. A multiplication in the frequency domain results in a convolution in the time domain. What's a convolution ? It's written x(t) * h(t) (with an asterisk) and is defined as this:

$$x(t) * h(t) = \int_{-\infty}^{\infty} x(t)h(t-t)dt$$

This means that, with an LTI system, an input spectrum is multiplied by the frequency response to give an output spectrum. For the same process in terms of time, the input signal is convolved with the impulse response to give the output signal.
Convolution gets used in signal and picture processing- it can generate 3D spacialised sound and sharpen up blurred images.

4 Correlation

Correlation is the process of measuring similarity- two waveforms correlate strongly if they are of similar shape. Thus, despite being 90 degrees out of phase, sinewaves and coswaves correlate strongly, because they are the same shape.
It's also possible to correlate a waveform against itself- this is called autocorrelation.

4.1 Autocorrelation and signal recovery

Autocorrelation is most commonly used in extracting periodic signals from random noise. The process of autocorrelation involves taking two copies of the waveform and then comparing them over a wide range of timeshifts. This is described mathematically by the following function:

$$r_{ff}(t) = \int_{-\infty}^{\infty} f(t) f(t+t)dt$$

This describes how f(t) matches itself as τ varies. Of course, you always get an exact match when $\tau = 0$. Of course, you can't normally integrate over an infinite period, but the longer your integration period, the better your correlation.

However, if you autocorrelate random noise over an infinite period, the function returns zero except when $\tau = 0$. Thus, you can identify a periodic wave hidden in noise by autocorrelating your signal. Any peaks in the autocorrelation function that aren't at $\tau = 0$ must be due to a periodic function hidden in the otherwise random noise.
Even better than that, a handy theorem allows you to find out exactly what's going on:

The Wiener-Khintchine Theorem says that taking the Fourier transform of your autocorrelation function will give you a power spectrum, allowing you to identify the signal easily. Your periodic signal will stand out well clear of the rest of the noise.

This is very useful- radio astronomers, who spend all their days looking for radio waves from distant stars and galaxies can identify periodic signals amidst the noise by autocorrelating their signals and then taking a Fourier transform. The longer the period you integrate over, the easier it is to separate signal from noise in the resulting power spectrum.

4.2 Pseudo-random binary sequences

Genuinely random noise is completely unpredictable and has components at every frequency. It's referred to as "white noise", because it contains every frequency, in the same way that white light contains every frequency of light. White noise is (more-or-less) what you hear when you tune a radio to a frequency where there's no transmission- just a continuous hiss.
You should remember measuring frequency responses of circuits from first year labs. It's very tedious- you tune the signal generator to each frequency in turn, and look at the resulting amplitude. This is called a frequency sweep. Now, suspend disbelief, and imagine that you have an infinite number of signal generators, each tuned to a slightly different frequency. You wire them all up in parallel, and drive the resultant wave into your circuit. Then look at the output on a spectrum analyser. What you should see is the frequency response of your circuit, because you've fed in every individual frequency. Since an infinite number of signal generators is rather hard to obtain, we can use a noise source instead. Noise contains every frequency- so it's equivalent to all those signal generators.
If you still find this hard to picture- imagine you are hosting a party. Your guests arrive slowly, so initially, only one person is talking at a time. However, as more people come, more and more people are talking simultaneously, and the overall sound becomes a seemingly constant cacophony! Despite the fact that you have hundreds of individual signals, their combined randomness generates noise.

However wonderful random noise is, it's very useful to have something that is superficially totally random, but is in fact 100% predictable. This is called a pseudo-random sequence. It's rather like a recurring decimal number- it's apparently random, until you reach the sequence length, at which point it repeats itself. The longer the sequence, the closer you get to being truly random.
The world of digital electronics lends itself to generating these sorts of sequences. In this situation, they're called pseudo-random binary sequences or PRBS. You can build a PRBS generator with a shift register and an XOR gate- there's lots of maths that describes how you wire them up, but usually you take two of the outputs of the shift register and feed them into the XOR gate. The output of the gate then gets fed into the shift register's input. This produces a PRBS that repeats every 2^n-1 clock cycles, where n is the number of stages in the shift register. So, for a common-or-garden 8-stage shift register, the generator will produce a seemingly-random sequence that continues for 255 clock cycles before it repeats.
The only drawback is the zero state- if the shift register initialises with all its stages set at logic 0, the generator locks up. This is easily corrected for- some logic can test for this condition, and force the input to the generator to be a logic 1.

4.3 Spread-spectrum communications

You can use a PRBS as a spreading code, to make your radio transmissions secure against people listening in. Unsurprisingly, this technology was originally used for military purposes. Your transmitter and receiver have synchronised PRBS generators, and you multiply your input signal (a digital datastream) by the PRBS and then transmit. At the receiver, you divide by the output of the PRBS generator and recover the original data. Clever, eh ? Anyone not knowing the PRBS being used will just receive apparently random noise. Civilian applications of this technique (known as spread-spectrum) include digital radio (DAB) and digital TV, which use PRBS to spread their data over a wide range of frequencies and increase the ruggedness of the transmission.

4.4 Cross-correlation and system identification

You can also use a PRBS for "system identification"- finding the impulse and frequency response of an LTI circuit. This technique is used by a lab instrument called a network analyser.

Basically, you feed your PRBS into your circuit under investigation, and then you compare the output with the PRBS you fed in. The most obvious way to do this is with a computer fitted with a data acquisition system: you feed the PRBS into the computer, whilst simultaneously feeding in the output from the circuit you're investigating. Then you can carry out a cross-correlation between the two signals. This involves time-shifting one and multiplying it by the other:

$$r_{xy}(t) = \int_{-\infty}^{\infty} x(t)y(t+t)dt$$

The result of doing this is an approximation to the impulse response- the response gets more accurate as the length of the PRBS gets longer.

In the lab, we used an analogue computer to do the cross-correlation for us, displaying the result on an oscilloscope. Because the analogue computer works in real-time and doesn't store its data, you have to generate a "mutant" PRBS, which has a one-clock delay in it at the end of every sequence. It gradually shifts with respect to the genuine PRBS. The true PRBS generator is fed into the circuit, and the output of the circuit is then correlated against the "mutant" by the analogue correlator. Over the time it takes for the "mutant" to go from being in-phase with the genuine PRBS to being in-phase with it again, the correlator produces a waveform resembling the impulse response.