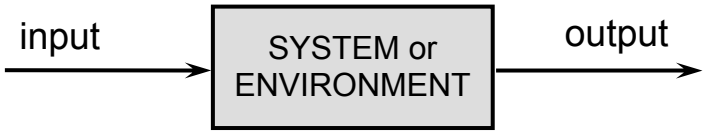
Chapter 1

TIME AND FREQUENCY

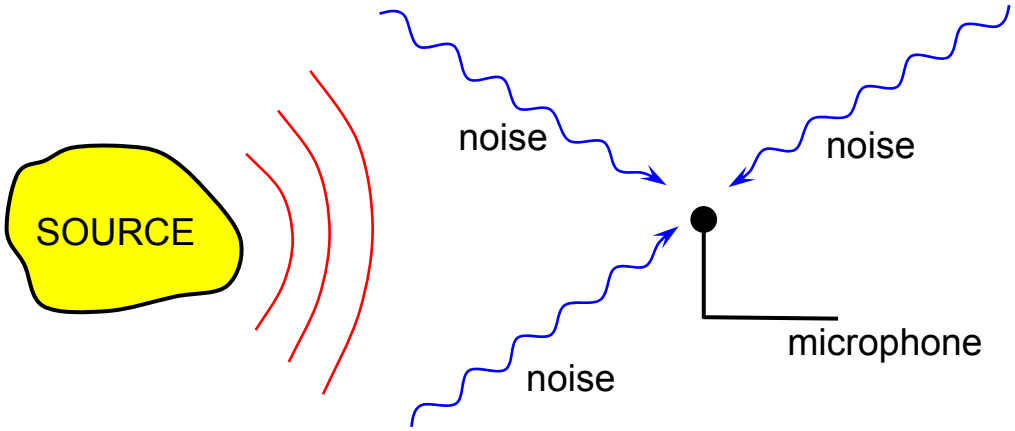
Introduction

This course is about signals and noise and, more specifically, how to measure what you want (the "signal") and reject what you don't want (the "noise"). There are many different kinds of measurements that you might make:

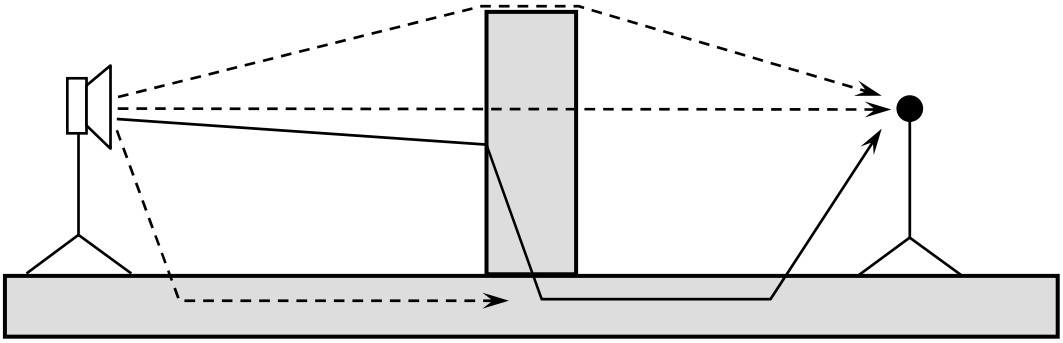
- You might want to discover the characteristics of a system or an environment by sending a signal in and measuring the result:



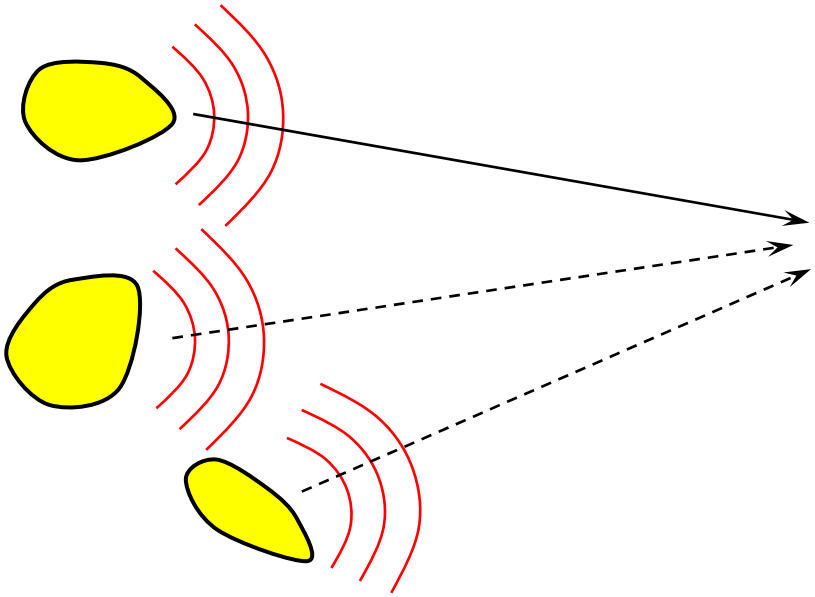
- You might want to determine the characteristics of a source of sound (or vibration) from measurements in a noisy environment:



- You might want to find the dominant path or paths that a signal takes:



- You might want to determine which of several noise sources contributes most to the noise somewhere else:



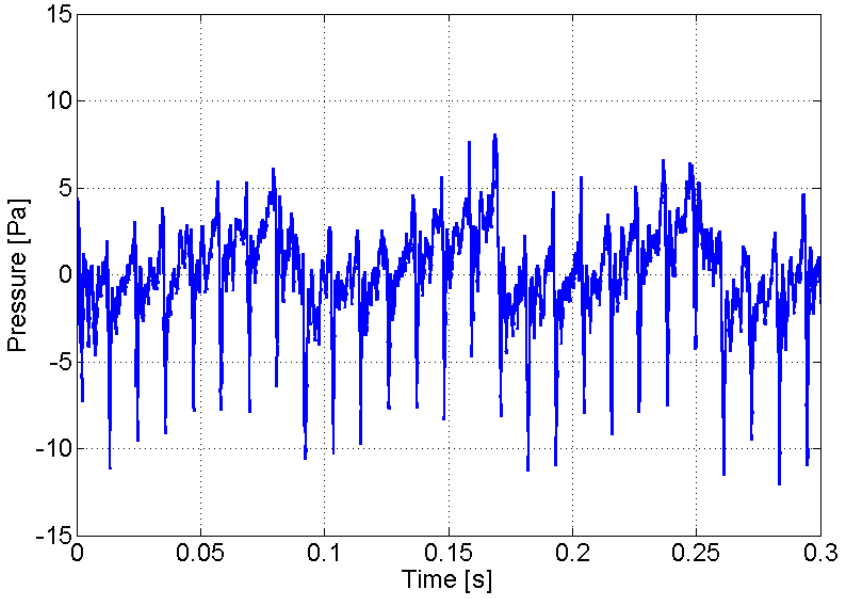
What can we measure? For an acoustic signal, we can measure the pressure variations of that signal using a microphone in air or a hydrophone underwater. For a vibrational signal, we can measure the acceleration of a surface. Normally, our equipment would convert the pressure or acceleration to a voltage and then would sample that voltage many times per second and store those samples as digital values. The result is a record of the variation of the measured quantity with time.

A great deal of information is in this time record and there are many tools for extracting this information. Most people have heard of the A-weighted decibel level (or dBA level) of sound. The A-weighted level is an attempt to collapse all of the information in an acoustic waveform into a single value that is, in some way, related to human perception of loudness. Many people have heard of dB or dBA but that's unfortunate: there isn't much information in that single number. If we measured the acoustic level during a music concert and then measured the acoustic level inside a dog kennel, we might find that the A-weighted sound level was identical in both places. You can imagine, however, that you might enjoy the concert more than lots of dogs barking. Because dBA is a single number, the content is lost; what remains is an artificial measure of "loudness" (whatever loudness is).

The tools that we have at our disposal are far more flexible than a single-number measure like dBA. We might ultimately want to find the A-weighted sound level but we can learn far more about the signals (and the noise) if we avoid the abstraction of a single number.

Much of this course covers various analysis tools. We might start with the acoustic pressure as a function of time. This record is, itself, an abstraction in that we use equipment to make the recording and that equipment does not have infinite bandwidth or infinite resolution in amplitude; however, by understanding the limits of our equipment and the signals we want to measure, we can capture the essential features.

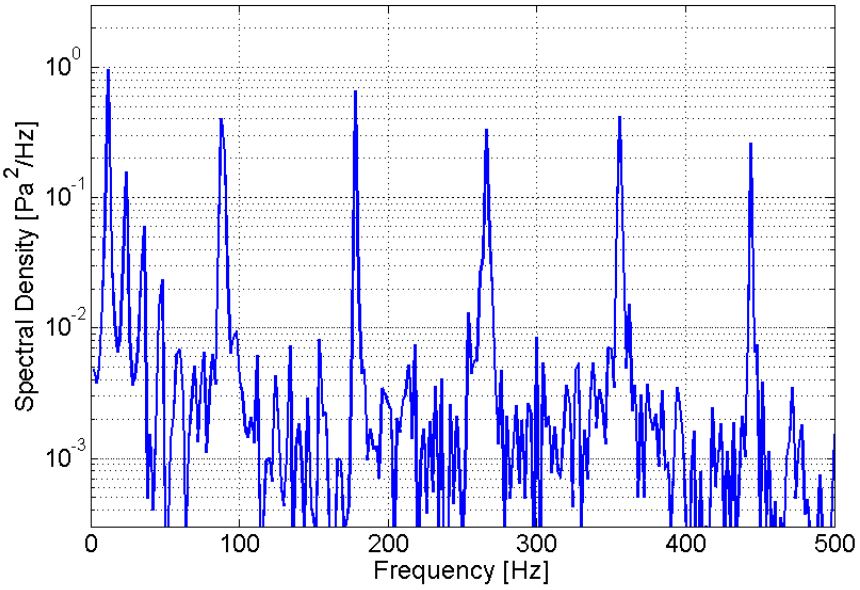
The plot below shows the acoustic pressure as a function of time for a helicopter making a low-level pass over a microphone. There are clear patterns in the time record.



Notice that the negative peaks are larger in absolute value than the positive peaks. Notice the periodicity of the negative peaks – there are about 18 peaks in 0.2 seconds – a rate of about 90 peaks per second. This is the "blade passage rate" of the helicopter's tail rotor. Notice the longer period with a sawtooth-like shape; that pattern repeats about every 0.08 seconds or about 12 events per second. This is the blade passage rate of the main rotor.

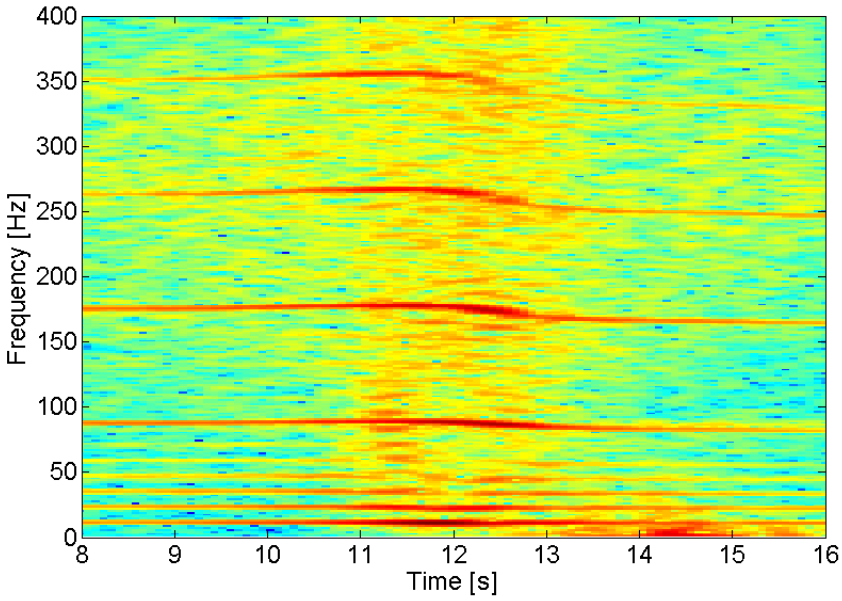
We can also examine this signal in the frequency domain by a transformation (the Fourier transform) that retains all of the original information (i.e., this transformation is completely reversible). By looking at the distribution of frequencies ("pitch"), we can isolate different components of the signal.

The frequency-domain transformation gives a series of complex numbers; if there are 1000 points in the time record, there will be 1000 complex values in the corresponding frequency record. If we know that we are only interested in the distribution of power over frequency, we can find the "spectral density," which is proportional to the magnitude-squared of the complex frequency-domain numbers. The plot below shows the spectral density for a 0.5 second interval of the helicopter flyover.



Here, we see a prominent peak at about 12 Hz with several "harmonics" – integer multiples at 24, 36, 48 Hz – and another prominent peak at 88 Hz with harmonics. This "line" structure in the spectral density reflects the main- and tail-rotor passage frequencies and the harmonics reflect the fact that the periodic pressure pulses are not simple sine waves. As we will see much later, an analysis like this can determine the number of blades in both the main rotor and tail rotor, the rotation rate of the blades, and even whether the engine is a turbine engine or a reciprocating piston engine!

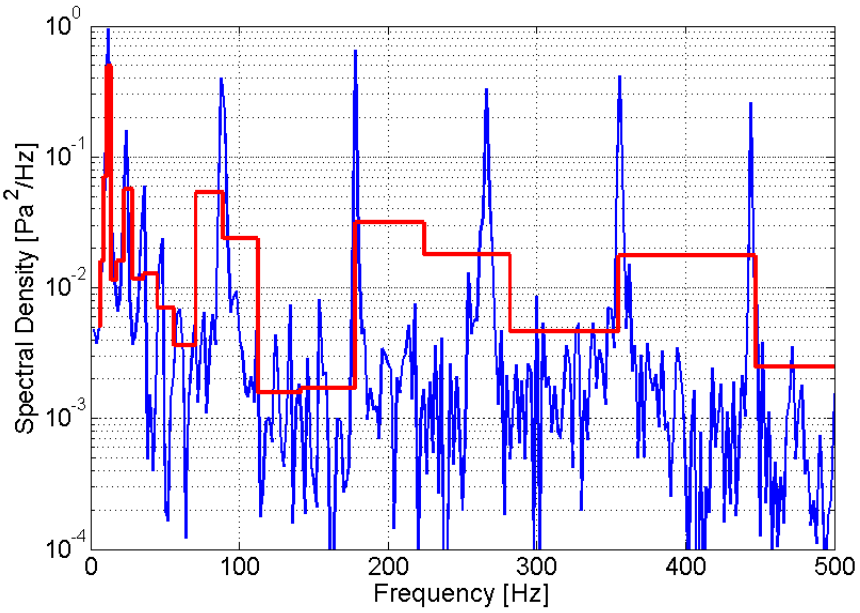
If we perform the frequency-domain transformation on the entire pressure time series, then any of the variation in time of the signal components is lost. We can, instead, perform the frequency-domain transformation on many short segments of the time series. The result is a two-dimensional matrix of values where one dimension extends over the range of frequencies and the other dimension extends over time. If we color code the magnitudes and plot on frequency and time axes, we paint visual picture (a "spectrogram", see Chapter 3) of the time evolution of the various frequency components. A spectrogram for eight seconds of the helicopter flyover is shown here.



This display tells more of the story. You can see the prominent frequency components as nearly horizontal lines. The spectral density is coded by color with the highest levels as red and the lowest as blue. The helicopter passed directly over the microphone at about 12 seconds and you can see the increase in level at the helicopter approaches the microphone and the decrease as it moves away. You can also see the Doppler shift in the frequency components (from which we can estimate the speed of the helicopter and the distance from the helicopter to the microphone at closest passage).

The frequency-domain transformation can produce fine resolution in frequency (i.e., resolving small differences in frequency from one point to the next) – the longer the time record, the finer the frequency resolution. In some cases, this might be too much detail. Alternatively, we can combine groups of adjacent frequency "bins" into wider bands – often into the so-called "octave bands" or "one-third-octave bands". By doing so, we lose detail in the frequency domain but we reduce the number of values that we need to examine. In the figure below, the one-third-octave spectral averages (red) have been plotted on top of the spectral density values[1] (blue).

[1] The one-third octave levels are shown as the average value of the spectral density in the particular one-third octave band. Normally, the integral of spectral density over the band is used. The average value, however, is more easily



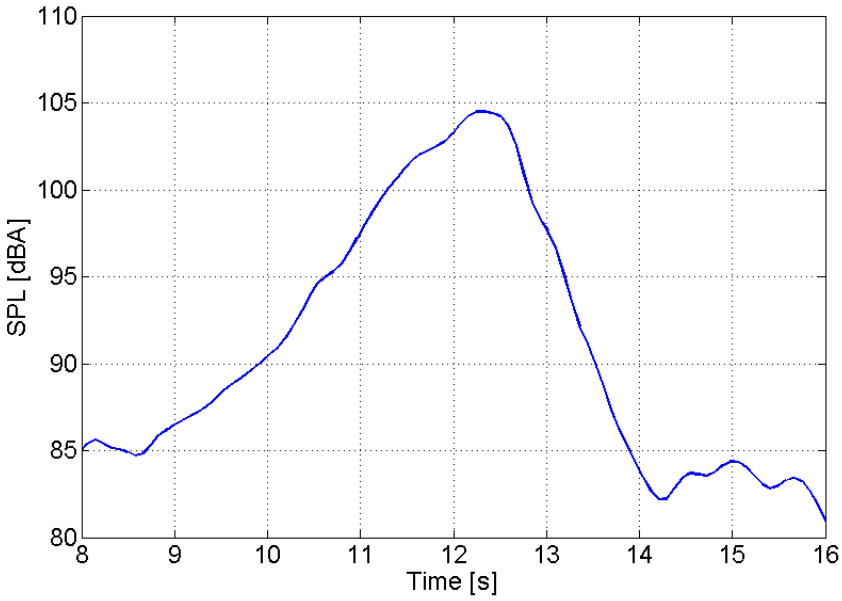
One-third-octave (or octave) bands are "proportional" bands – the width of each band is a fixed percentage (about 23% for one-third octave and 71% for octave) of the band's center frequency. Notice that the steps in the red curve are wider at higher frequency. Notice also that the harmonic structure (except at the lowest frequencies) is lost.

If we combine all of the frequency values while weighting each value according to the average human perception of loudness across frequency, the result is a single number for the time record. This is the origin of the A-weighted sound level. Here, all details of the frequency content are lost. From the spectrogram, we would be able to tell the difference between a concert and a dog kennel; we might even be able to tell the difference looking at the one-third-octave band levels; but we have no hope of discriminating based on the A-weighted levels.

We can retain some of the time dependence by segmenting the time series (taking every second, for example) and computing the A-weighted level for each segment. The following plot shows the time variation of the A-weighted level for the helicopter passage.

---

compared to the spectral density (blue curve). The more conventional band level is the average value times the width of the band in hertz.



The increase in level followed by a decrease is clear; however, there is no longer any indication of the pressure fluctuations associated with the blade rotation. In fact, we might see a similar curve from the passage of a large truck or even from a stationary fire-alarm siren that spins up and then spins back down. This curve is the equivalent of the output of a sound-level meter set for A-weighting and "slow" response.

In some cases, we might want to abstract even further and develop a single number that represented an average A-weighted level (of, say, many aircraft flights) over 24 hours. Now, both the frequency content and the temporal content are irretrievably lost.

While it is important to understand how to construct single-number metrics as most legislation regarding acceptable noise levels and most studies of human perception of sounds are based on these metrics, the field of acoustic measurement and analysis is far richer than these severe abstractions. Consequently, most of this course deals with more detailed analyses while also developing the tools to understand the single-number metrics. *If all you learned from a course in signal measurement was how to operate a sound-level meter, you would know almost nothing about acoustic measurement.*

There are a few concepts you should know before starting this course. You should be familiar with

- basic acoustic, electrical, and mechanical quantities and their units;
- simple integration and differentiation;
- complex numbers and their components – magnitude and phase or real and imaginary parts, and complex-exponential time dependence ($e^{j\omega t}$).

If you are comfortable with any computer programming language or analysis software, you should be able to translate the concepts in this course into working software; however, much of what is written here is oriented toward MatLab or Octave[2] as a computational tool.

Classical analysis of random data

This course is not a course in analysis of random data; this is a course in analysis of real measurements, the components of which invariably have some degree of randomness.

Often the theory of random data analysis starts with "ensembles" – many measurements of "identical" processes where the measurements are taken simultaneously. "Expected Values" are taken to find averages of these ensembles. However, acoustic or vibration measurements of many simultaneous, identical processes are rare. Typically, a single process is measured (perhaps by many sensors at *different* locations) over some period of time.

The transition from the well-developed statistics of ensembles to measurements of a single process extended in time is often done by assuming that

[2] This statement is not intended as an endorsement of MatLab or Octave (an open-distribution language almost identical to MatLab) but these are convenient tools for a course like this. Do not, however, relinquish control to any software package – understand what you are doing and make sure you control the software rather than letting the software control you.

the process is "stationary" – this is its statistics are not changing with time. In practice, many (most?) real measurements are of processes that are clearly not stationary – measurements of ambient noise, or of the noise of cars passing by, or aircraft landing and taking off from airports. Machinery noise may not change as quickly but it still is subject to changes in operating conditions and the changes may be even more important than the periods of steady operation.

If a process output has the characteristics of a random variable with a Gaussian distribution and is stationary, then there is a large body of applicable theory regarding analysis of noise and the errors associated with various measures of those random processes. This is the domain of classical random data analysis and this domain is well-addressed by the classics texts by Bendat and Piersol[3]. At times, we will borrow relevant results from random-data analysis but always with the understanding that our measurements will probably not have the characteristics of ideal random processes.

Typical signals and measurements in acoustics and vibration

Many real measurements are taken over relatively short periods of time. The concept of taking limits as the time period goes to infinity needs to be replaced by recognition of the actual duration of the data record. Random components are often not stationary. Furthermore, errors in measurement may be dominated by noise in measuring equipment rather than by the fundamental statistical errors associated with measuring ideal random variables.

Real signals are often limited in duration, unsteady in amplitude, and are often poorly behaved in statistics. A few signal types:

- short-duration signals – like explosions;

[3] Bendat and Piersol, *Random Data: Analysis and Measurement Procedures*, 3rd Ed., Wiley, 2000; or Bendat and Piersol, *Engineering Applications of Correlation and Spectral Analysis*, 2nd Ed., Wiley, 1993.

- long-duration, periodic signals – like tones, which can vary in amplitude and in frequency; and
- long-duration, random signals – like "noise", which can also vary in amplitude and vary in frequency distribution.

Keeping these realities in mind, we will focus on those aspects of classical analysis that are useful for the sorts of measurements normally encountered in acoustics and vibrations work. Some of the methods operate in the frequency domain and other methods operate in the time domain.

The key elements in analysis are:

- the time series (e.g., acoustic pressure as a function of time),
- the linear spectrum (a complex function of frequency),
- the spectral density (a distribution of signal power over frequency).

Much of the discussion in this text starts with one or more of these three elements and we will spend considerable time exploring them.

We will also consider techniques to make the experiments and measurements better. When faced with measurements with high variability, we could simply analyze the error and accept the inherent uncertainty. A far better strategy[4] is to improve the measurement! Fortunately, there are a number of powerful techniques for reducing errors and improving signal-to-noise in measurements. It is important to understand the classical techniques of random data analysis but it is also useful to understand that, by careful design of an experiment or a measurement, we may be able to make the desired signal much easier to "see" in the inevitable background of instrument or environmental noise.

[4] It is sad commentary that there are *many* more books on statistics than there are on measurement technique.

One of the most fundamental operations is the transformation from the time domain to the frequency domain (and back again) so we will start there and build on that base.

Time Domain and Frequency Domain

The Fourier transform takes us either from the time domain to the frequency domain or from the frequency domain to the time domain. The integral form of the transform from the time domain to the frequency domain (the "forward" transform) is

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi f t}\, dt. \qquad \text{(I-1)}$$

Here, $X(f)$ is a complex function of the frequency, f. $X(f)$ is the **linear spectrum**[5] of the function of time, $x(t)$. The function, x, would be the measured voltage or pressure or acceleration (or digitizer units or wav file units). However:

- We never take data for an infinite length of time. We only know x for some finite length of time, T.
- We don't normally have continuous functions. Our equipment (or the subsequent translation to a computer file) samples the measured function usually at evenly spaced intervals, Δt.

Consequently, we need to create a version of the transform that is suited to finite, sampled **time series**[6].

[5] Pay attention to the names of various functions and be precise in your use of technical language. "Linear spectrum" is not universally accepted for $X(f)$ but the term serves to distinguish clearly from "spectral density" defined later.

[6] We'll call the sampled voltage, pressure, or acceleration a "time series" to emphasize that it is in the time domain and that it is sampled.

Finite discrete Fourier transform

If we have evenly spaced points in both the time and frequency domains, then the continuous variables, t and f, are replaced by discrete values:

$$t \rightarrow n\,\Delta t \qquad \text{and} \qquad f \rightarrow m\,\Delta f\,. \tag{I-2}$$

(I'll try to be consistent in using n as the index for the sampled time series and m as the index for the sampled spectrum.) If we have N samples of the time series ($T = N\Delta t$), then we can convert the integral transform,

$$X(f) = \int_{-\infty}^{\infty} x(t)\, e^{-j2\pi f t}\, dt \quad ,$$

into a sampled equivalent:

$$X(m\,\Delta f) = \sum_{n=0}^{N-1} x(n\,\Delta t)\, e^{-j2\pi (n\Delta t)(m\Delta f)}\, \Delta t\,. \tag{I-3}$$

We've replaced the integration by a summation (over the finite time "record"), we've replaced the continuous integrand by its sampled version, and we've replaced the differential, dt, by its sampled equivalent, Δt. (Don't forget the differential! If you leave it out, the units of $X(f)$ and $X(m\Delta f)$ would be different!)

To condense the notation, let

$$x_n = x(n\,\Delta t) \qquad \text{and} \qquad X_m = X(m\,\Delta f)\,. \tag{I-4}$$

While the functions of time will almost always be real numbers, the linear spectrum values, X_m, will in general be complex.

Discrete linear spectrum

By forcing the following relationship,

$$\Delta f \cdot \Delta t \equiv \frac{1}{N}, \tag{I-5}$$

extremely efficient algorithms can be designed to compute these transforms. The summation then takes the form of the discrete linear spectrum:

$$X_m = \sum_{n=0}^{N-1} x_n \, e^{-j2\pi \frac{nm}{N}} \, \Delta t \quad . \tag{I-6}$$

We will perform this particular calculation often in this course. Except for the Δt factor, the right side of this equation is the Discrete Fourier Transform (DFT), which is implemented in most software packages as the Fast Fourier Transform (FFT). In MatLab or Octave[7], the '*fft*' function is used. (Type '*help fft*' in the command window for specifics.)

There are two critical points concerning calculation of the linear spectrum:

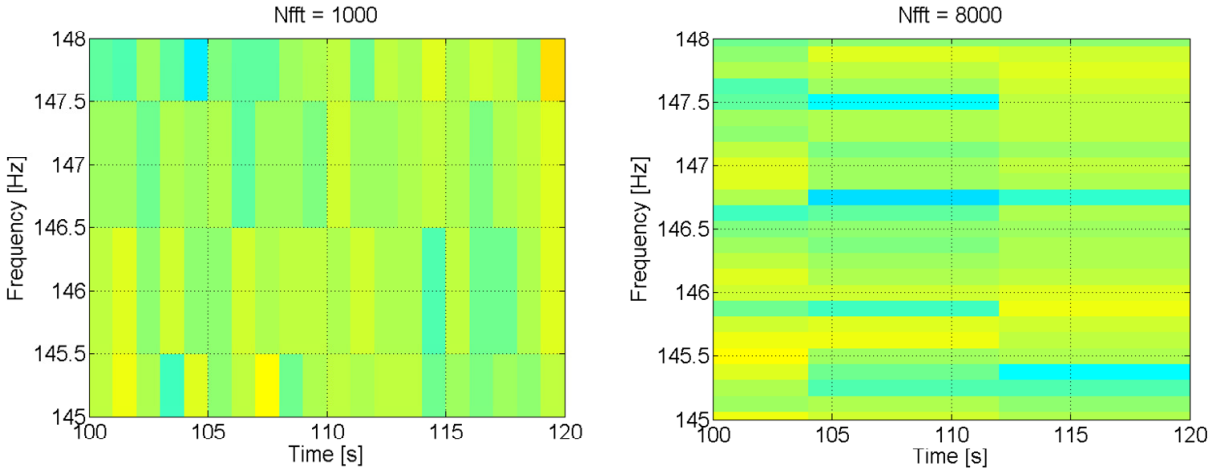
- Because of the relationship, $\Delta f \Delta t = 1/N$, once we've chosen a sampling rate ($f_s = 1/\Delta t$) and a certain number of points, N, in the time record, we have **absolutely no choice** about the spacing, Δf, of spectrum points in the frequency domain.
- The output of the *fft* function is **NOT** the linear spectrum. You must include the Δt factor.

The relationship, $\Delta f \Delta t = 1/N$, has serious consequences for spectrum analysis. Since $N\Delta t = T$, where T is the length of the record, this relationship can also be written at $\Delta f = 1/T$. The only way to improve the resolution in the

[7] If you are using another software package like MathCad or Mathematica, *you must check* to see just what that package is calculating when you use their fft's. They use slightly different forms in their fft's.

frequency domain (i.e., to decrease Δf) is to increase the duration of the record, which, for the same total number of points, *decreases* the resolution (makes the time steps larger) in the time domain. A finer-scale examination of the frequency content results in a coarser look at the time variations and vice versa.

One way of illustrating the consequences is through the spectrogram – a display built from successive spectral density[8] calculations where the value of the spectral density is coded by color and, for a single time record, the color pixels are oriented in a column. The pixels for the spectral density of each successive time record are oriented in adjoining columns with the start time of each record increasing to the right. The figure below shows small sections of two spectrograms for the same function of time. In both cases, the sample rate is 1000 samples per second so Δt is fixed (at 0.001 s). The display on the left is for $N = 1000$ (i.e., one-second records or $T = 1$ s); the display on the right is for $N = 8000$ (i.e., eight-second records). Only a small section of the pixilated display is shown.



On the left, the resolution in time (1 second) is much finer than on the right (8 seconds) so the display on the left would be able to show faster time variations in the signal than the display on the right. However, the frequency resolution on the right (1/8 Hz) is much finer than that on the left (1 Hz). The display on the right could resolve much finer frequency structure. Increasing the resolution in one dimension decreases the resolution in the other dimension.

[8] The "spectral density" is defined later. For now, consider it to be proportional to the signal power at each frequency in the linear spectrum.

This is a classic limitation in frequency-time analysis and there is not much that we can do about it other than being constantly aware of the trade-off. Much later, we will discuss time-domain filtering, which does not depend on the Fourier transform at all; however, the narrower in bandwidth these filters are, the slower they respond to changes in the input – an analogous trade-off.

Symmetry in frequency

Normally, time-series data are series of real numbers—often the output values of an analog-to-digital converter. When the time series is real, there is a specific symmetry in the linear spectrum. Consider the linear-spectrum value for a negative m (a negative frequency index):

$$X_{-m} = \sum_{n=0}^{N-1} x_n \, e^{-j2\pi \frac{n(-m)}{N}} \Delta t \quad . \tag{I-7}$$

The only change on the right side is in the exponent of the complex exponential factor. If the time series, x_n, is real, then the complex conjugate of the entire right side would also only affect the exponent. Consequently (for real x_n),

$$X_{-m} = X_m^* \quad . \tag{I-8}$$

In other words, the linear spectrum (if we pretend that we can extend the frequency index outside the "normal" range of 0 to N-1), is complex-conjugate symmetric about $m = 0$.

Also notice that adding N or any integer multiple of N to m makes no change in the resultant linear spectrum (whether x is real or not) as this is simply equivalent to adding 2π to the exponent of the complex-exponential factor:

$$X_{m\pm N} = X_m \quad , \tag{I-9}$$

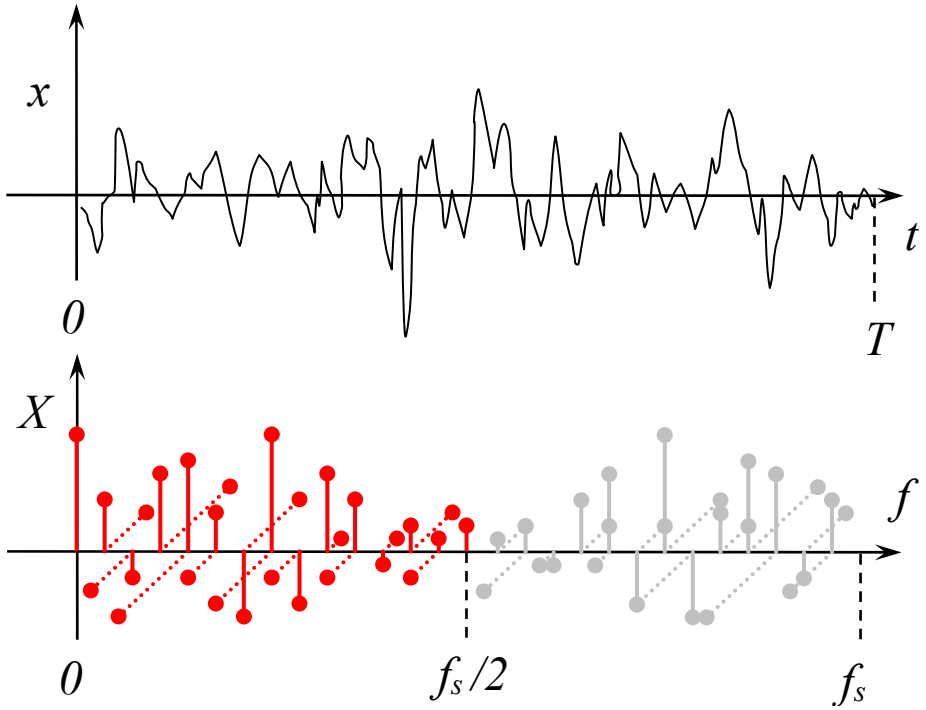
which demonstrates that the linear spectrum (if examined for frequency indexes beyond the 0 to N-1 range) repeats periodically.

For a real time series, there is also complex-conjugate symmetry in the linear spectrum about the $m = N/2$ point. If p is the index offset from $N/2$, then

$$X_{\frac{N}{2}+p} = X_{\frac{N}{2}-N+p} = X_{-\frac{N}{2}+p} = X^*_{\frac{N}{2}-p} \quad . \tag{I-10}$$

While the linear spectrum has the same number of points as the time series from which it was derived, almost half of the linear-spectrum points are mirrored by their complex conjugates (if the time series is real, rather than complex).

The transformation from the time domain to the frequency domain, then, takes N time-domain points and generates a complex spectrum that is also N points long; however, for a real time series only $N/2 + 1$ of those complex points are unique[9]:



[9] You might wonder how N points are compressed into $N/2+1$ points. Keep in mind that a single complex number has two components—a real part and an imaginary part. If we add up all the parts, there actually are N "pieces" in the unique part of the linear spectrum (two of those points are entirely real as we'll discuss later).

The sampled time series (*N* real points, upper plot) is transformed to the complex linear spectrum (N complex points, lower plot). The complex values are indicated by the solid line segments (real parts) and by the dotted line segments (imaginary parts). The complex values in gray are equal to the complex conjugates of the values in red (between and **not including** the points at 0 and $f_s/2$) and they appear in the reverse order—they *mirror* the points below $f_s/2$.

Inverse transform

By a similar conversion of the integral form of the inverse Fourier Transform to a sampled form, we find that the inverse discrete transform is,

$$x_n = \sum_{m=0}^{N-1} X_m \, e^{j2\pi\frac{nm}{N}} \, \Delta f \quad . \tag{I-11}$$

The MatLab '*ifft*' function includes a factor of $1/N$; to reconstruct the right side of the equation above, you must multiply the output of the *ifft* function by N and by Δf. Note that the product of N and Δf is equal to the sampling frequency so multiplication by f_s (or dividing by Δt) is equivalent. Without these factors, the MatLab *ifft* of a sampled linear spectrum does **NOT** produce the original time-domain function.

In MatLab[10], then, the forward transform is

$$X_m = fft(x_n)\,\Delta t \quad , \tag{I-12}$$

and the inverse transform is

$$x_n = ifft(X_m)/\Delta t \quad . \tag{I-13}$$

[10] If you are converting this discussion to another software package – Mathematica or MathCad, for example – be sure to find out exactly how the *fft* and *ifft* are implemented in those packages. The details are different and the appropriate adjustments must be made to find the true linear spectrum and the true equivalents for the forward and inverse transforms.

There is a convenient "symmetry" to the above expressions: the forward transform is the MatLab *fft* times Δt; the inverse transform is the MatLab *ifft* divided by Δt.

In writing equations, I will use the convention that the indexes, *m* and *n*, run from 0 to *N*-1 for *N* total points. MatLab[11], however, assumes that its vectors are indexed starting from 1, not 0, so you'll have to adjust for this when programming in MatLab. While the index, *n*, runs from 0 to *N*-1, the MatLab variable would be indexed from 1 to *N*. In the expressions below for $n = 1$, 2 and 3, the MatLab version is shown on the far right:

$$\begin{aligned} x_0 &= x(t=0) = x(1) \\ x_1 &= x(t=1\cdot\Delta t) = x(2) \\ x_2 &= x(t=2\cdot\Delta t) = x(3) \quad , \end{aligned} \tag{I-14}$$

and so on. Index the linear spectrum in a similar manner—the first point in the linear spectrum in MatLab order is the zero-frequency point ($m = 0$; MatLab index = 1).

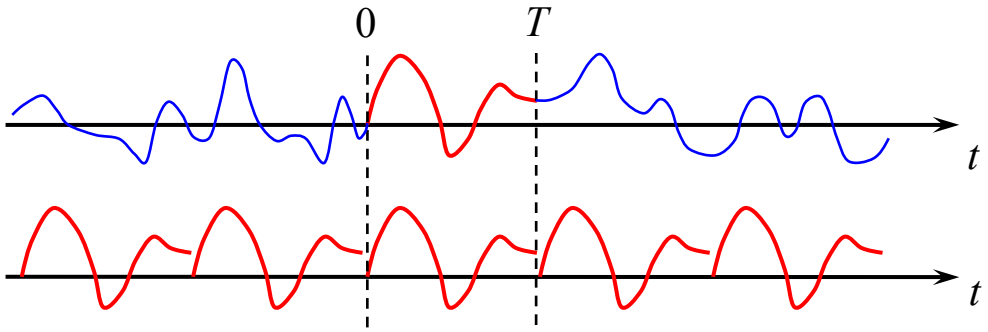
Implied periodicity and symmetry

The complex exponential factor, $e^{\pm j2\pi nm/N}$, is the operator (the "kernel") in either the forward transform or the reverse transform. Adding or subtracting any integer multiple of *N* to either *n* or *m* is equivalent to adding or subtracting a multiple of 2π in the exponent so the result is unchanged. Consequently,

$$x_{n+N} \equiv x_n \qquad \text{and} \qquad X_{m+N} \equiv X_m \quad . \tag{I-15}$$

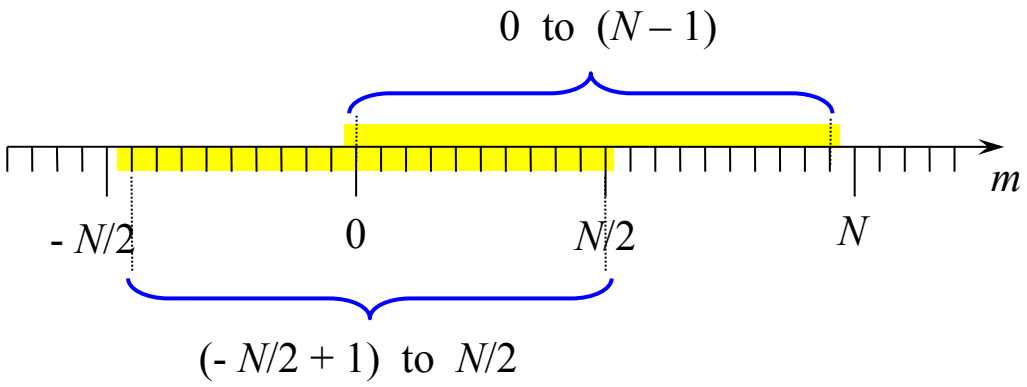
[11] Know your software! Python, for example, indexes vectors and matrices from zero, not from one. This is consistent with the mathematics but can be confusing when transitioning from MatLab to Python or from Python to MatLab. Octave indexes as MatLab does.

The process of sampling in both the time and frequency domain is equivalent to assuming that both the time series and the linear spectrum are repeated over and over from negative infinity to positive infinity. Instead of picturing the transform of an isolated time period from 0 to T seconds, you could consider that you're taking the transform of that pattern repeated every T seconds. In the figure below, the top trace is the physical time series and we are interested in the transform of the segment from 0 to T. The bottom trace is the periodically repeated equivalent time series.



Notice that the repeated sequence has discontinuities every T seconds. The linear spectrum is completely defined by N points (actually by only $N/2 + 1$ points if you consider the complex-conjugate symmetry) but it, too, is effectively repeated every $N\Delta f$ hertz in the frequency domain.

Since both x_n and X_m repeat every N points, we could take *any* consecutive string of N points in the periodically repeated series to represent the sampled function. In the frequency domain, for example, we could take the set of points from 0 to N-1 or the set from $-N/2+1$ to $N/2$:



Recognize, though, that the *fft* function returns the 0 to *N*-1 version in which the first value ($m = 0$ or MatLab index = 1) corresponds to zero frequency. (Similarly, the *ifft* function returns samples and the first sample corresponds to $t = 0$).

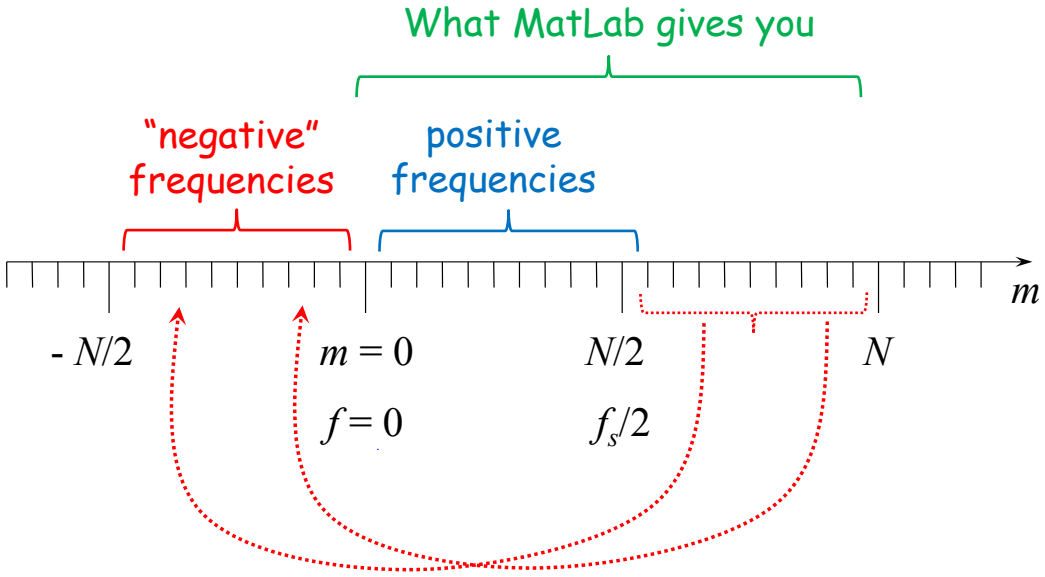
The upper set shown in the figure above is the set that MatLab returns, namely,

$$0 \le m \le N-1 \qquad \text{or} \qquad m = 0,\ 1,\ \ldots\ N-1,$$

(with the MatLab index running from 1 to *N*). The lower set,

$$-\frac{N}{2}+1 \le m \le \frac{N}{2} \qquad \text{or} \qquad m = 0,\ \pm 1,\ \pm 2,\ \ldots\ \pm\left(\frac{N}{2}-1\right),\ \frac{N}{2},$$

is often used to show the symmetry about zero frequency for the linear spectrum. In this set, the points for $m < 0$ are called "negative-frequency" points—these points can be interpreted as having negative frequencies equal to $m\Delta f$:



The frequency, $f_s/2$, is a special frequency. Called the Nyquist frequency, it is the frequency about which the positive and negative frequencies show the complex-conjugate symmetry in the linear spectrum. Also, only frequency

components *below* $f_s/2$ can be represented accurately by the sampling and transformation operations described above.

Note: We've assumed an even number of points (as would be most common[12]) but notice a consequence of that choice. Because one point is at zero, there is no way to pick the block of samples so that it extends as far below zero as it does above. We have $N/2$ samples on one side of zero and $N/2 - 1$ samples on the other side. (If N were odd, then we could pick the block of samples to be symmetric about zero.)

We will often find it convenient to work with the smaller, unique set of X_m where $m = 0$ to $N/2$. Because of their complex-conjugate symmetry with corresponding values for m less than zero, this half-spectrum (actually, N/2+1 values) makes up the so-called "positive-frequency" spectrum.

Please be careful, though. Remember that MatLab always generates X_m for $m = 0$ to N-1 (with the MatLab array index running from 1 to N) so the "negative-frequency" spectrum values are **above** the positive-frequency portion in the output vector. Also remember that, for N even, you must keep an odd number ($N/2+1$) of values to completely define the spectrum.

[12] Working with an odd number of points is not difficult, though. See later in this chapter for details.

Exercise 1.1: The Linear Spectrum

This exercise may seem trivial but please try a few examples with MatLab's *fft* function. Generate an 8-point vector with a "random" selection of numbers – just type in some arbitrary sequence of numbers. Call that vector the time series. Since you are inventing the time series, pick some Δt. That sets the sample rate, f_s. The number of points and Δt set the duration of the time record, which, in turn, sets the frequency increment, Δf.

Find the linear spectrum. Print out the values of the linear spectrum and study the pattern. What values are real? What pairs of values are complex conjugates? Determine the frequencies that correspond to each of the values of the linear spectrum. What point corresponds to 0 Hz? What point corresponds to $f_s/2$ Hz?

Now try this again but make all of the points in the original "time" series the same number. What do you observe? Try again but make the points in the time series the same value but alternating in sign. What do you observe? You can often learn a great deal about complicated functions by performing simple tests.

Discrete mathematics and intervals

If I took vacation from the 13th through the 15th of October, how many days was I away from work? Three days: the 13th, 14th and 15th. We do discrete math correctly all the time; however, it's easy to make mistakes when not working with things as concrete as vacation days. You might argue that this period is only two days long (15 minus 13 is 2) but you'd be wrong. Wrong because "day" in this context represents a 24-hour period, not an instant of time. This is discrete math.

In similar fashion, you should think of the samples of a time series as representing intervals Δt long rather than as instantaneous points in time. Suppose you wanted to analyze a 10-second period starting at 43.0 seconds in a recorded data stream. Suppose this data stream is sampled at 10 samples per second so Δt is 0.1 seconds. If you thought that the 10-second period runs from 43.0 to 53.0 seconds, you would take 101 samples: the first sample at 43.0 and the last sample at 53.0. But this is incorrect! Each sample represents a full Δt interval. The period corresponding to 101 samples is (101 samples)*(0.1 seconds per sample) or 10.1 seconds. For sampled math, the 10-second period would be 100 samples long. The center point of the first sample would be at 43.0 seconds and the center point of the last sample would be at 52.9 seconds. If you then took the next 10-

second period, the first sample's center point would be 53.0 seconds, exactly 10 seconds from the first sample of the preceding period.

If you think of each sample as representing an interval, you will be fine. One hundred 0.1-second intervals equals 10 seconds. (Don't cheat yourself out of any vacation days!)

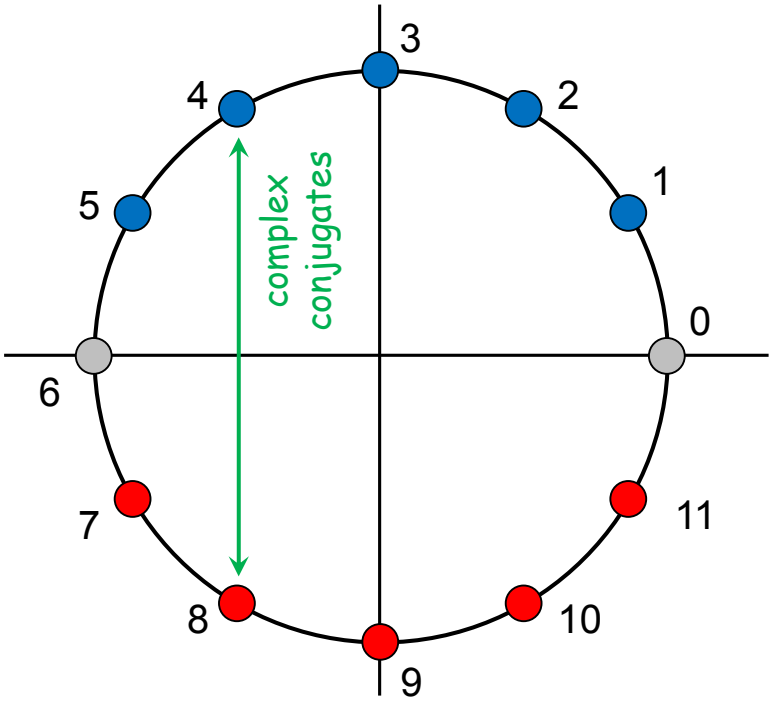
The unit circle: a visual aid

During the transition from the continuous Fourier transform to the discrete linear spectrum, we imposed the condition, $\Delta f \Delta t = 1/N$. As a result, the complex exponential in either the forward or reverse transform takes on a simple form that is a function of the integer indexes, m and n. The inverse transform, for example, is

$$\frac{1}{\Delta t}\sum_{m=0}^{N-1} X_m \, e^{j2\pi\frac{nm}{N}} \quad . \tag{I-16}$$

In the full transform where we would want to compute all N values of x_n, there would be N^2 exponents to calculate; however, because of the form of the exponent, there are actually only N *different* values of that exponent. This is the principal reason for the exceptionally high efficiency of FFT algorithms.

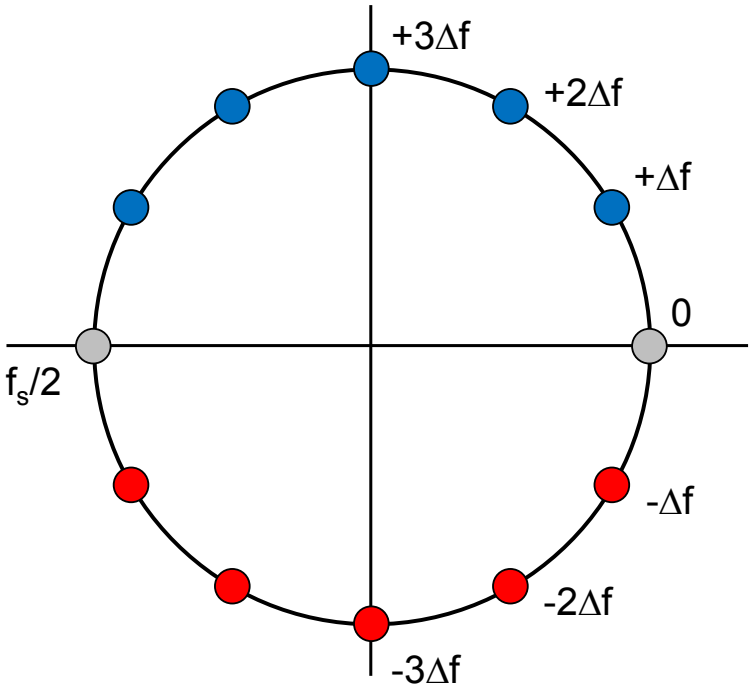
By plotting the values of the complex exponential (not just the exponent but the entire exponential factor) in the complex plane, we can get some insight into the structure of the transform. The N distinct points plot along the unit circle in the complex plane:



In the figure above, *N* is 12 and the numbers are the values of the frequency index, *m*. Notice that points 0 and N/2 (6 in this example) are real. Points that have the same real value (for example, points 4 and 8) are complex conjugates—I've shown these sets as blue and red sets. Each blue point has a red complex-conjugate mate. An important point is that, regardless of the value of *N*, *the N points are evenly distributed around the circle*. If *N* is even, there is always a point on the negative real axis; if *N* is odd, there is not; however, the complex-conjugate symmetry is always there.[13]

This unit-circle representation gives a graphic view of the arrangement of points. While the unit circle in the complex plane is often used for describing *z*-transforms and system stability that is not the intent here. I'm introducing the unit circle here simply as a visual aid to understanding the distribution and character of the points of the linear spectrum. Labeling the points by their frequencies is also illuminating:

[13] As long as the time series is real rather than complex. We will examine the special applications of complex time series but the results of a physical measurement will be a real time series.



The zero-frequency point is on the positive real axis; frequency increases (in steps of Δf) counter-clockwise around the upper half circle; and frequency decreases (in steps of $-\Delta f$) clockwise around the lower half circle. If N is even, then one point of the linear spectrum is at the Nyquist frequency ($f_s/2$), which is the point on the negative real axis. If N is odd, there will not be a point at the Nyquist frequency.

The highest positive frequency represented on the diagram is the Nyquist frequency. If there are frequency components above the Nyquist frequency, their contributions are *aliased* onto other points in the diagram. We will investigate aliasing shortly—for now, consider that the points in the diagram can represent the frequencies as labeled in the figure above but any point can also represent a frequency of the labeled frequency plus any integer multiple of the sampling frequency. If the actual frequency component is, for example, $f_s + 3\Delta f$, then its contribution will be aliased onto the "$3\Delta f$" point. If the actual frequency component is, for example, $f_s - 2\Delta f$, then its contribution will be aliased onto the "$-2\Delta f$" point. Aliasing puts energy in the wrong place in the frequency domain so we will take steps to avoid it.

Note: The complex values on the unit circle are *not* the values of the linear spectrum; they are simply the values of the transform kernel (the complex exponential factor) at the corresponding frequencies.

Exploring the FFT

So far, I've been working through the classic derivations of properties of the transforms; however, derivations quickly induce boredom. Simply exploring the characteristics of the transforms by "playing" with them can be effective in learning how to use the transforms and how they behave. We'll take this exploratory approach in this section. These are exercises for you to do using MatLab (or Octave or Python or your analysis software of choice).

In each case, use a small number of points (6 to 12) so that you can inspect each point in the time series and in the transform. For these exercises, you can ignore the factor of Δt in the linear spectrum as long as you promise to remember that you've ignored it! Observe the results of the *fft* function on the various time series—where are the real values, where are the complex values, what can you deduce about the relationship between the time series and the transform for these cases?

simple, random time series

If you haven't done Exercise 1.1, do it now.

slowest possible variation

The slowest possible variation a time series is no variation at all. Construct a time series of all ones and examine the result of using that time series as input to the *fft* function.

fastest possible variation

The fastest possible variation in the time series is if the values alternate between positive and negative values. Construct a time series of 1, -1, 1, -1, 1, -1,… and examine the corresponding *fft* output.

sines and cosines

If we convert the continuous cosine function,

$$x(t) \quad = \quad A\cos(2\pi f_0 t) \quad , \qquad \text{(I-17)}$$

to its discrete equivalent (and impose the familiar $\Delta t \Delta f = 1/N$ condition), we have

$$x_n \quad = \quad A\cos\left(2\pi \, \frac{m_0 n}{N}\right) \quad ; \quad n \quad = \quad 0, 1, \ldots N-1 \quad . \qquad \text{(I-18)}$$

If m_0 is an integer, the resulting time series contains an integer number of full cycles of a cosine with frequency $m_0 \Delta f$.

For $A = 1$ and $N = 16$, generate the time series of a cosine with $m_0 = 3$. Examine (or plot) the time series. How many cycles of oscillation are there? (Remember discrete math: the time series will start at the positive peak (one, because $A = 1$) but it will end one sample short of another positive peak. The periodic extension of that time series will continue starting with a positive peak value. If you stitch several replicas together, the combination will also have an integer number of cycles.) Try a few other values of m_0. Where is (are) the non-zero value(s) in the linear spectrum? Are they where you would expect them to be (comparing the time-domain pattern to the implied frequencies of the *fft* output—you can pick a specific sample rate if it helps you visualize this comparison). Try this for a sine wave instead of the cosine wave. What changes in the *fft* output?

aliasing

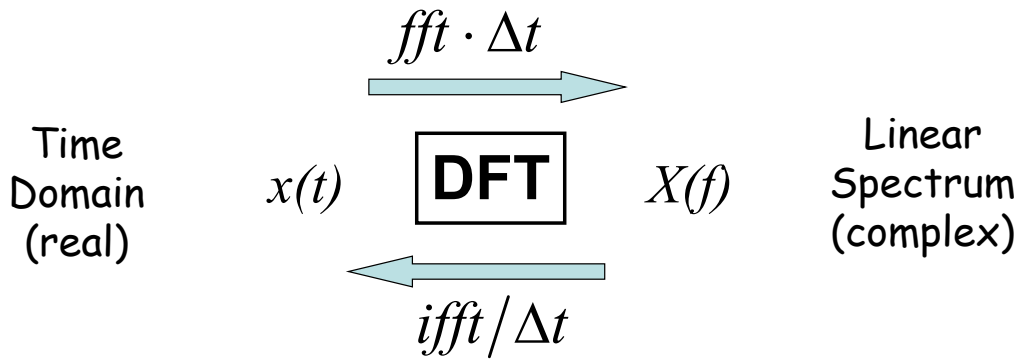
Using the expression for the discrete cosine function, generate a time series using a value for m_0 that is equal to $N/2$. Where is (are) the non-zero value(s) in the *fft* output? Now try a value for m_0 that is greater than $N/2$. Where is (are) the non-zero value(s) in the *fft* output? Plot the time series. What frequency does it appear to have? To illustrate more clearly, you could plot a cosine curve with the same frequency but a much higher sample rate and the original sampled version on the same plot. Try a few values for m_0 greater than $N/2$. Can you write an expression that will predict where the non-zero values will appear for any m_0 value?

non-integer number of cycles

Using the expression for the discrete cosine function, generate a time series using a value for m_0 that is not an integer. Where is (are) the non-zero value(s) in the *fft* output? There will probably not be any non-zero values. Make several copies of that time series and join them end-to-end. Plot this result. Notice the discontinuities at the boundaries between copies. Those discontinuities generate what's often called spectral leakage. The signal power spreads outside the "bin" that "contains" the selected frequency.

Review

We now have the most basic tool for time/frequency analysis: the discrete transform pair. We can go back and forth between the time series, $x(t)$, and the linear spectrum, $X(f)$, and the transforms are reversible. Normally, the time series will be real and the linear spectrum will be complex with a particular symmetry.



The transforms, forward and inverse, are as follows:

$$X_m = \sum_{n=0}^{N-1} x_n \, e^{-j2\pi\frac{nm}{N}} \, \Delta t = fft\left(x_n\right)\Delta t \quad , \tag{I-19}$$

$$x_n = \sum_{m=0}^{N-1} X_m \, e^{j2\pi\frac{nm}{N}} \, \Delta f = ifft\left(X_m\right)/\Delta t \quad , \tag{I-20}$$

along with a number of basic relationships between parameters,

$$f_s = 1/\Delta t \quad ; \quad T = N\,\Delta t \quad ; \quad \Delta f = 1/T \quad , \tag{I-21}$$

where f_s is the sampling frequency in samples per second (or hertz), Δt is the time between samples in seconds, T is the duration in seconds of the time series, N is the number of points in the time series, and Δf is the frequency interval in hertz between values of the linear spectrum.

Units

Physical units are important both for understanding systems and processes and for finding errors. Many papers and texts in digital signal processing normalize to non-dimensional quantities early. I understand the power of working with non-dimensional quantities—I often work with normalized variables myself; however, you should not normalize until you are sure you understand what you are

doing. Preserving physical units is one of the tools that are useful for gaining this understanding.

Consequently, I will try to use units as often as possible even if it means making up ("assuming" as the intelligencia would say) units. So far, I have mentioned the possibility that our time series might be in pascals[14] for an acoustic pressure function or acceleration units for an accelerometer but I have not carried those units much further. With modern data-acquisition hardware, the most common data quantity is the output of an analog-to-digital converter. This makes assigning a unit awkward. Use something that makes sense to you. I often label these values as DU (digitizer units). By giving every quantity a specific unit, you can trace processes from end to end and pick up mistakes in conversions or calibrations.

For example, suppose we have data from a 16-bit digitizer with a full-scale input voltage range of ± 5 volts and we want to find the voltage equivalent for a digitizer value of 2564 DU. The total range of digitizer units for a 16-bit digitizer is 2^{16} DU or 65 536 DU. Since the voltage range is a symmetric plus/minus range, it's more convenient to use the equivalent plus/minus in DU or ± 32 768 DU[15]. Consequently, the voltage-to-DU conversion is 5 volts per 32 768 DU or 1.5259e-4 V/DU. Then (2564 DU)*(1.5259e-4 V/DU) gives 0.3912 volts. The units check and the voltage is reasonable. It's far too easy to make a mistake if you assume that the digitizer output is dimensionless—that prevents you from checking units.

As you progress through this text, you should work out the units of each new quantity as that quantity is defined. The simplest way to work out units is usually from an equation that involves the quantity. For example, to find the units of linear spectrum, start with the defining equation:

---

[14] At some point, you may be inclined to point out that pascal is a proper name and should be capitalized. However, in the SI system of units, the convention is that if the unit is a person's name, it is not capitalized unless the name is abbreviated. Hence, pascal or Pa and hertz or Hz are all correct whereas Hertz and hz are not. I didn't make the rules.

[15] If you object, you would be correct. A span of ± 32 768 would include both positive and negative end points and zero making a total number of values one too many for the 16-bit system. The actual range for normal digital arithmetic is from -32 768 to + 32 767.

$$X_m = \sum_{n=0}^{N-1} x_n \, e^{-j2\pi \frac{nm}{N}} \, \Delta t \quad . \tag{I-22}$$

If the time series, x_n, is in pascals, the units of the linear spectrum will be pascal seconds (Pa s) because the complex-exponential factor is dimensionless and the units of Δt are seconds. (Because seconds and 1/Hz are dimensionally equivalent—both are seconds—the units may also be written as pascals per hertz or Pa/Hz.)

Caution regarding analysis software packages

Analysis packages like MatLab often have a wide variety of special-purpose functions. It never hurts to be skeptical of these functions. In this text, we will use only a few of the special-purpose functions. These include *fft*, *ifft*, and *filter* (I'll describe *filter* at length later) as these functions have been optimized highly and tested extensively. They also have few options, which makes using them simpler and less error prone. We will also use plotting functions extensively; it would be too much work to write plotting routines from scratch.

In this course, we will cover other functions like spectral density, correlation and coherence, and construct spectrograms. MatLab (and other packages) has special functions for these operations; however, you will not understand what's happening unless you take the time to write your own routines to do these operations. Also, the MatLab versions are difficult to use properly. They often have many optional inputs and the defaults are often not good choices. It can take you as long to understand fully these packaged functions as it does to write your own and writing your own is invaluable for developing your understanding.

Constructing a time series with a specific frequency distribution

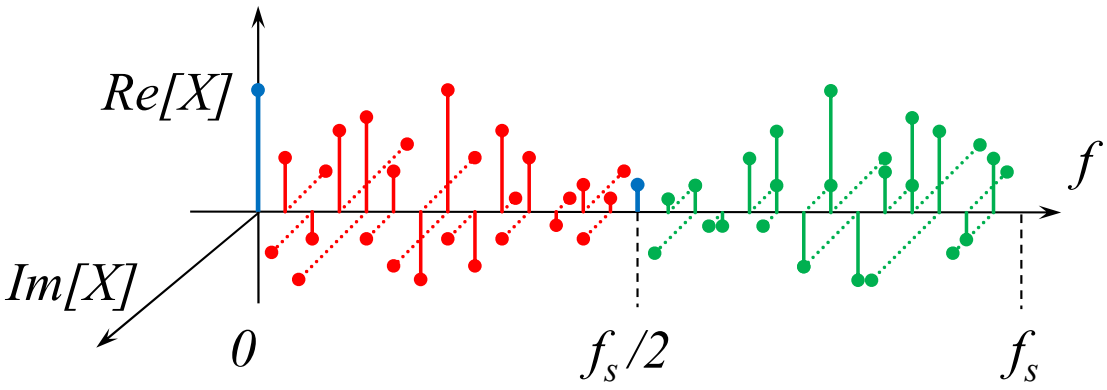
A great deal has been written about random "signals" with uniform distribution of power over frequency and a Gaussian distribution in amplitude. The expression, white noise, is used to indicate the uniform distribution over frequency. The well-developed body of theory for such random functions makes analysis of processes and systems easier than for more general signals; however, white, Gaussian noise is not as common as many texts and papers would lead you to believe. Ambient noise, whether indoors, outdoors, or underwater, is rarely white—there is often a strong frequency dependence of the power distribution. Real sources of sound or vibration also often have significant spectral shaping. Testing a process, an algorithm, or a system with artificial white noise may yield misleading results when the actual process or system will normally be exposed to noise with a shaped spectrum.

Fortunately, it's relatively easy to generate a time series with a specific frequency distribution but that also "looks like" noise. You may know the spectral distribution of the expected noise (see spectral density in the next major section of this chapter); you may even be able to measure the noise in question. This gives you an estimate of the magnitude of the linear spectrum.

If you know the magnitude of the linear spectrum at all of the relevant frequencies (zero through $f_s/2$), you can construct a full linear spectrum. In order to produce a real time series, the linear-spectrum points at zero frequency and $f_s/2$ must also be real. It is often acceptable to set these two points to zero but, in any case, you must make those values real. Then take the remaining positive-frequency points and make them complex by introducing a random phase. The phase should be uniformly distributed over the 0 to 2π range. Having both magnitude and phase allows construction of complex linear-spectrum values over the positive-frequency region. Then construct the "negative" frequencies by making the values above the $f_s/2$ point complex conjugates of the values below the $f_s/2$ point. If this construction is done correctly, the inverse transform of the linear

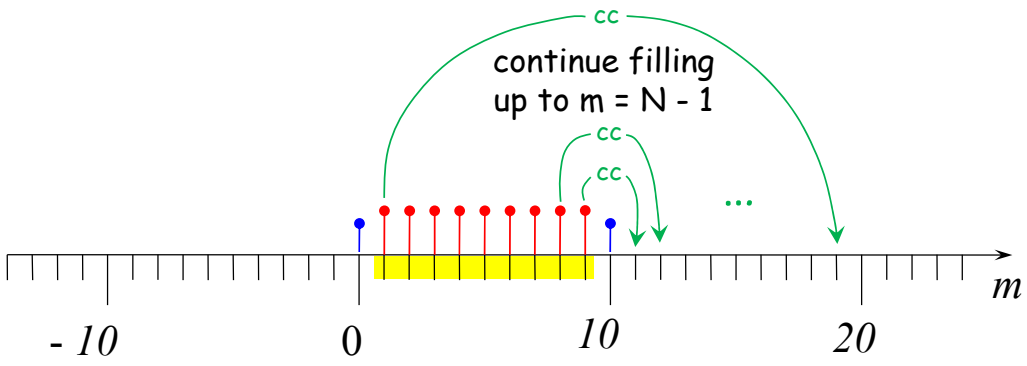
spectrum will be a real-valued time series[16] with the irregular appearance of noise. Each new selection of a set of random phases will generate a new time series with the same spectral distribution of power.

The following diagrams illustrate proper construction of the linear spectrum. The first figure is a stem plot of the real (solid lines) and imaginary (dotted lines) parts of the linear spectrum. The points at zero frequency and $f_s/2$ (blue) have no imaginary parts. The "negative-frequency" points (green) are constructed from the positive-frequency points (red) by complex-conjugate symmetry about the $f_s/2$ point. (Notice that there is no point at f_s—that would give one too many points in the linear spectrum.)

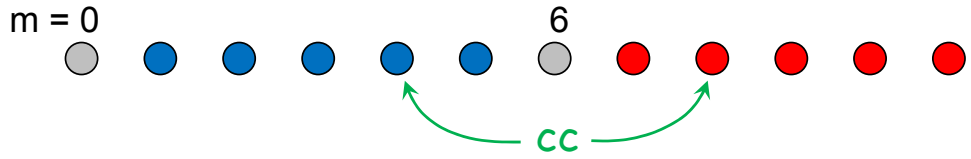


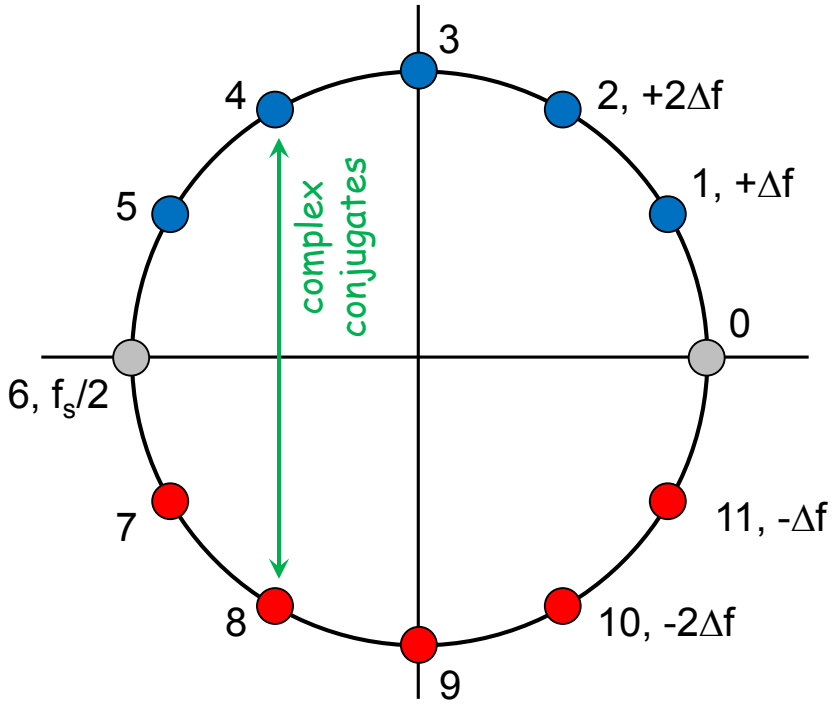
The next figure, an example with $N = 20$, shows more clearly the mirroring of the positive frequency values into the negative-frequency region (cc stands for complex conjugate).

[16] Round-off error in computer arithmetic can lead to small imaginary parts. These artifacts will be far smaller than the corresponding real parts in a properly constructed time series.



The last figure in this sequence shows the relationship between the symmetry for a linear spectrum in MatLab order (top) and the unit-circle representation (bottom) for a case in which $N = 12$. The symmetry from point 4 to point 8 is highlighted (green arrows).





As we've already discussed, for even N and a real time-domain result, you must specify $N/2+1$ points to determine completely the spectrum. The positive-frequency spectrum will extend from $f = 0$ ($m = 0$) to $f = f_s/2$ ($m = N/2$) where f_s is the sampling frequency. The quantity, $f_s/2$, is sometimes called the Nyquist frequency. In order to generate the linear spectrum in the form expected by the *ifft* function, we must generate another $N/2-1$ points (the "negative" frequencies although they must be appended *above* the "positive" frequency points).

Keep a few issues in mind. This complex conjugate construction is only valid if we know that the time-domain function is real. While that is the case most often in this course, don't try to use this construction if the time-domain function is complex[17]. Also, the $N/2+1$ spectrum points for $m = 0$ to $m = N/2$ correspond to the frequencies from $f = 0$ ("DC") to $f = f_s/2$ where f_s is the sampling frequency. (Remember that $f = m\Delta f$. Furthermore, f is the *center* frequency of the "bin".)

Finally, remember that if you're constructing a complex spectrum in order to generate a time-domain waveform, you must recognize that the spectrum points at $m = 0$ and $m = N/2$ are real. Even if you are developing the spectrum values from some analytical expression that gives a complex result at $f = f_s/2$, you must force the value to be real or the time-domain result may have a significant imaginary component. It is often safe enough to set that value to zero (and, if you need to, you can taper the last few spectral points before the $m = N/2$ point to avoid a sharp change in spectral magnitude). If you set the $m = 0$ point to zero, you'll generate a time function with zero mean value, which may, in fact, be desirable.

Make sure you test the time-series result. Round-off error in the computer arithmetic may produce a very small[18] imaginary part even though, theoretically, the time series should be real. While it's tempting to simply take the real part of the inverse transform of the linear spectrum, don't include this step in your code

[17] We never expect a complex time series from a physical measurement; however, there are circumstances in which it is useful to work with artificially complex time series. An example of this in connection with the Hilbert transform is considered later in these notes.

[18] Comparison word and phrases are dangerous: "very small" compared to what? If you don't give a basis for the comparison, you fail to communicate. Is $4.2\text{x}10^{-14}$ very small? Not in relation to $4.2\text{x}10^{-35}$! Always be suspicious when you see "small", "fast", "large", and so on without any frame of reference.

until you have verified that any imaginary part generated is, in fact, tiny compared to the real part[19]. If there is a significant imaginary part in the time series, chances are you've done something wrong.

Exercise 1.2: Time Series from a Linear Spectrum – "Perfect" Random Noise

Create a random time-domain signal eight points in length by generating a linear spectrum and then transforming that linear spectrum to the time domain:

Set the *magnitude* of the points in the linear spectrum, X_m, from $m = 0$ and $m = N/2$ to one. (The MatLab indexes would run from 1 to $N/2+1$.) Make the *phase* of each X_m random with a uniform distribution between 0 and 2π (but set the phases of X_0 and $X_{N/2}$ equal to zero to keep those values real). This produces the positive-frequency linear spectrum. Generate the other part of the spectrum as described in the text and join the two parts together. Check that you've done this correctly! Make sure that the total length is correct and that the values and their complex conjugates are in the proper locations.

Use *ifft* to find the time-domain signal. The output of *ifft* may be complex. Display a few of the values. The imaginary parts should be zero or VERY small. (They may not be zero because of internal round-off errors in the MatLab computations.) If the imaginary parts are not extremely small relative to the real parts, there's an error somewhere. If the imaginary parts are very small, then take the real part of the result to save as the time-domain waveform.

Once you are sure that you are creating a proper linear spectrum (one that produces a real time series), change the number of points to 1000. Plot the resulting time series. Does it look random? Calculate the mean. Calculate the mean-square value of the time waveform and the root-mean-square (*rms*) value.

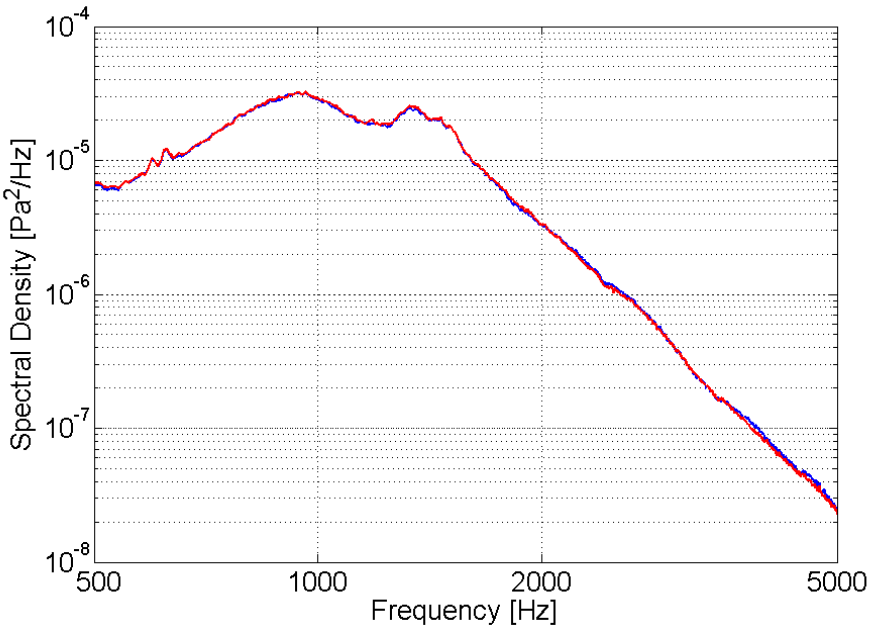
Re-do the generation process but this time make all of the phases zero. Make the same plots and calculations. Notice that the two time-domain waveforms look nothing alike even though the two waveforms have exactly the same magnitude of their linear spectra.

Generation of a time series that has a particular spectral shape is not simply an academic exercise. With this process (see the Exercise above), we can create a time series with a perfectly flat spectrum ("perfect white noise") or we can generate perfect pink noise (a spectral power density proportional to $1/f$, which we can generate by making the magnitude of the linear spectrum proportional to one over the square root of frequency). Beyond these "ideal" signals, we can also create time series that mimic actual background noise.

[19] Here's the frame of reference: tiny *compared to the real part*.

Often signal processing algorithms and systems are tested with signals embedded in white noise because it is easy to generate white noise. However, if the actual noise spectral density is markedly different from white noise (as it often is), then such tests can give misleading results. If the spectral distribution of actual background noise is known, we can generate a time series with that spectral shape and perform a far more realistic test of the process.

For example, if we wanted to develop a method to detect defective mufflers on an interstate highway, that process would have to function in the background of normal highway noise. The spectral density[20] shown below was measured near an interstate highway:



The background noise is certainly not flat over frequency (i.e., it is not "white") and the region above 1000 Hz drops much more steeply than pink noise. We can use the square root of this spectral density as the magnitude of a linear spectrum, make the phase random, and transform to the time domain. The result would be a time series in acoustic pressure that has the same spectral distribution as the actual highway noise. With each new set of random phases, we get another time series.

[20] I've mentioned "spectral density" several times already. We'll develop this quantity in the next section. For now, just consider that spectral density describes the distribution of power over frequency.

By using these artificial time series, we can mimic the expected background noise when testing detection algorithms. This technique is not well-known but it is powerful.

Signal "Power" in Time and Frequency

By now, we can take a sampled time series, x_n, of length $T = N\Delta t$,

and generate its linear spectrum, X_m, with $\Delta f = 1/T$. If the time series values are real, then the so-called "negative-frequency" components of the linear spectrum can be calculated from the positive-frequency components; only half (actually $N/2+1$) of the spectral components are unique. So far, so good but we can add to our analytical capability; we can use the linear spectrum to develop a quantitative measure of the distribution of power over frequency. This leads to the critically important "**spectral density**".

Consider the "power" in a signal. If the signal is, for example, a voltage, the instantaneous power into an electrical resistor, R, is

$$\boldsymbol{P}_{inst} = e(t)\cdot i(t) \quad , \tag{I-23}$$

but, the current, i, into a resistor is the voltage, e, divided by R. Therefore,

$$\boldsymbol{P}_{inst} = \frac{e^2(t)}{R} \quad . \tag{I-24}$$

The average power (over some length of time, T) is

$$\boldsymbol{P}_{avg} \quad = \quad \frac{1}{R}\frac{1}{T}\int_0^T e^2(t)\,dt \quad . \tag{I-25}$$

The quantity to the right of the $1/R$ factor is the "mean-square" voltage – the average value of the square of the instantaneous voltage. If R is independent of time, then the mean-square voltage is directly proportional to the average power. (In other words, for constant R, the average power is proportional to the mean-square voltage; the value of R may or may not be important depending on the application.)

Mean-square and root-mean-square

The mean-square value of a function is critically important in signal analysis. The mean-square relates to power. Furthermore, the mean-square gives us a quantitative bridge between the time domain and the frequency domain—an extremely valuable tool for checking results. Since we work with sampled functions, we can replace integrals with sums (as long as we also include the proper differential). Consequently, the mean-square value of x_n is

$$ms[x_n] \quad = \quad \left\langle x_n^2 \right\rangle \quad = \quad \overline{x_n^2} \quad = \quad \frac{1}{T}\sum_{n=0}^{N-1} x_n^2\,\Delta t \quad , \tag{I-26}$$

where three alternate symbols for mean-square are shown. Also, since $T = N\Delta t$, this expression is equivalent:

$$\overline{x_n^2} \quad = \quad \frac{1}{N}\sum_{n=0}^{N-1} x_n^2 \quad . \tag{I-27}$$

It is essential that you become comfortable with the mean-square. The expression, root-mean-square, is also used: it is simple the square root of the mean-square. Notice that the units of the mean-square are the units of x_n squared.

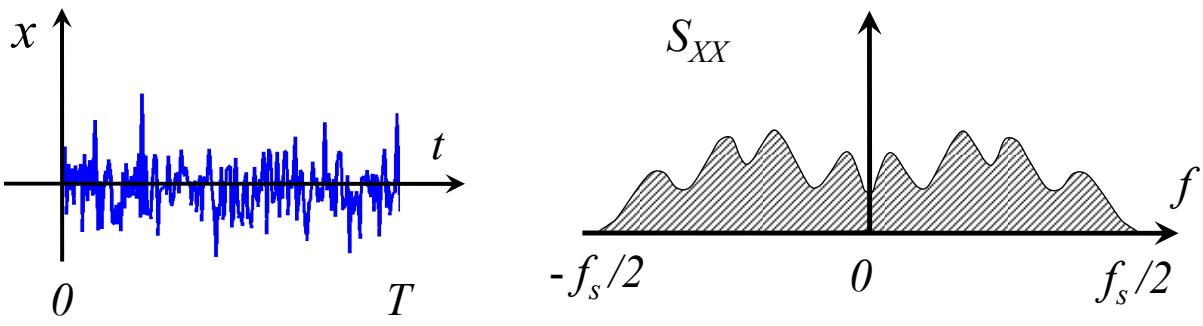
Distribution of power in the frequency domain

Instead of working exclusively with the complex linear spectrum, let's invent a function of frequency that describes the distribution of "power" (i.e., mean-square voltage or acoustic pressure) in the frequency domain. We want this power distribution function to be defined so that the integral (which we'll compute[21] as the sum of samples times Δf) over all frequency is equal to the mean-square value that we would have calculated in the time domain. Or, if we just integrate over a limited range of frequencies, the result will equal the mean-square value in the time domain if we had passed the signal through a filter that allowed just that range of frequencies to pass.

While it's more intuitive to consider only positive frequencies, development of this power-distribution function is simpler is we start by considering the entire linear spectrum—both positive and negative frequencies. We'll call this power-distribution function the power-spectral density[22] or, more specifically, the double-sided power spectral density to acknowledge that we're including both positive and negative frequencies. In the notation of Bendat and Piersol, this quantity is S_{XX}. We want the following relationship to be true:

[21] In this course, we will almost always replace integrals by sums but never forget to include the differential Δf in this calculation.

[22] Of course, it's not power; it's mean-square voltage (or pressure or acceleration). But this quantity is commonly called power spectral density instead of voltage-squared spectral density. Bendat and Piersol prefer "auto-spectral density" or "auto-spectrum." I will use simply "spectral density".



mean-square value of x_n = integral of S_{XX} from $-f_s/2$ to $f_s/2$

Note that, in their sampled forms, both x_n and S_{XX} have the same number of points.

As integrals of continuous functions, the connection between the time and frequency domains is

$$\frac{1}{T}\int_0^T x^2(t)\,dt \quad = \quad \int_{-f_s/2}^{f_s/2} S_{XX}\,df \quad . \tag{I-28}$$

To construct this spectral density function, we'll use Parseval's Theorem[23]. For continuous functions, if $x(t)$ and $X(f)$ are Fourier transforms of each other, the "energy" in the time domain equals the "energy" in the frequency domain:

$$\int_{-\infty}^{\infty} x^2(t)\,dt \quad = \quad \int_{-\infty}^{\infty} \left|X(f)\right|^2 df \,. \tag{I-29}$$

If we select a finite length time "record" from 0 to T seconds and if we sample the time and frequency functions, then we can write the sampled-signal version of Parseval's Theorem:

$$\sum_{n=0}^{N-1} x_n^2 \cdot \Delta t \quad = \quad \sum_{m=0}^{N-1} \left|X_m\right|^2 \cdot \Delta f \quad . \tag{I-30}$$

[23] Labeling functions, theorems, and the like by people's names lends an artificial sophistication and hides the underlying meaning. You should know the names but it is far more important to understand the principles. Parseval's Theorem is simple in concept; don't be put off by the name.

(Remember that X_m is the linear spectrum, not simply the output of an *fft*; the output of the *fft* function must be multiplied by Δt in order to become the linear spectrum. Also, another way of writing the magnitude-squared of X is X^*X. In MatLab, the first way could be written, abs(X).^2, and the second way, conj(X).*X.)

We want to create S_{XX} so that it obeys our condition: we want the integral of S_{XX} over all frequency ($-f_s/2$ to $f_s/2$) to be equal to the mean-square value of x_n. The mean-square (*ms*) value of x_n is

$$ms\left[x_n\right] \;=\; \frac{1}{T}\sum_{n=0}^{N-1} x_n^2 \cdot \Delta t \;\equiv\; \frac{1}{N}\sum_{n=0}^{N-1} x_n^2 \quad , \tag{I-31}$$

and we want the mean-square of x_n to be equal to the integral of S_{XX} from $-f_s/2$ to $f_s/2$. Expressing this integral as the sum of the sampled spectrum values (times Δf), we have the defining relationship for S_{XX}:

$$\frac{1}{N}\sum_{n=0}^{N-1} x_n^2 \;=\; \sum_{m=0}^{N-1} S_{XX} \cdot \Delta f \; . \tag{I-32}$$

Using the sampled-signal version of Parseval's Theorem, we can rewrite the left side in terms of the linear spectrum,

$$\frac{1}{N}\sum_{n=0}^{N-1} x_n^2 \;=\; \frac{1}{N}\frac{1}{\Delta t}\sum_{m=0}^{N-1}\left|X_m\right|^2 \Delta f \;=\; \frac{1}{T}\sum_{m=0}^{N-1}\left|X_m\right|^2 \Delta f \, , \tag{I-33}$$

so the relationship for S_{XX} becomes,

$$\frac{1}{T}\sum_{m=0}^{N-1}\left|X_m\right|^2 \cdot \Delta f \;=\; \sum_{m=0}^{N-1} S_{XX} \cdot \Delta f \; . \tag{I-34}$$

We can ensure that this relationship is always satisfied by defining S_{XX} as

$$S_{XX} \equiv \frac{1}{T}\left|X_m\right|^2 = \frac{1}{T} X_m^* X_m \quad . \tag{I-35}$$

The function, S_{XX}, is the **double-sided spectral density** (or double-sided auto-spectral density). S_{XX} has N values and all of the values are real.

Since S_{XX} is defined in terms of the absolute value squared of the linear spectrum, there is absolutely no additional information in the negative-frequency values: in the linear spectrum, the negative-frequency values are complex conjugates of positive-frequency values; in the spectral density, the negative-frequency values are identical to the positive-frequency values. Consequently, we can construct the more intuitive "**single-sided spectral density**", G_{XX}, where G_{XX} exists only for positive (and zero) frequencies:

$$G_{XX} \equiv \frac{2}{T}\left|X_m\right|^2 \qquad \text{for} \qquad 1 \le m \le \frac{N}{2} - 1 \quad , \tag{I-36}$$

over values of m for which there are corresponding points in the negative-frequency spectrum and

$$G_{XX}\Big|_{m=0} \equiv \frac{1}{T}\left|X_0\right|^2 \qquad \text{and} \qquad G_{XX}\Big|_{m=N/2} \equiv \frac{1}{T}\left|X_{N/2}\right|^2 \quad , \tag{I-37}$$

for the two end points of the positive-frequency spectrum. (Remember that X_0 and $X_{N/2}$ are both real but there's no harm in using the absolute-value operation.)

To convert from S_{XX} to G_{XX},

$$G_{XX} = 2\,S_{XX} \quad \text{for} \quad 1 \le m \le \frac{N}{2} - 1 \quad , \tag{I-38}$$

and

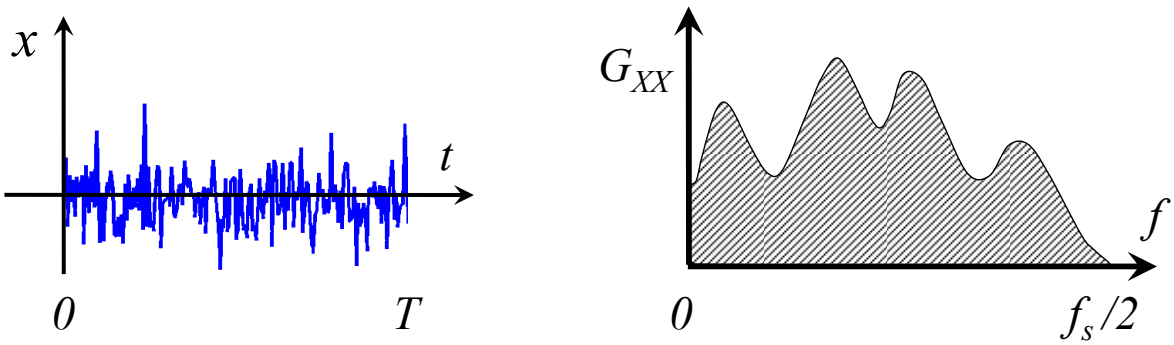
$$G_{XX} = S_{XX} \quad \text{for} \quad m = 0,\ N/2 \quad . \tag{I-39}$$

Think back to the unit-circle viewpoint. There is a value of S_{XX} for each point on the sampled unit circle. The negative-frequency values are on the half circle below the real axis. For the linear spectrum, these values are the complex conjugates of the corresponding values on the half circle above the real axis; whereas, for S_{XX} all values are real and corresponding values above and below the real axis are equal. What we're doing in constructing the single-sided spectral density is simply adding corresponding points above and below the real axis. From another viewpoint, double the points above the real axis and discard the points below the real axis. And the points on the real axis we leave alone because they do not have corresponding partners.

It's inconvenient that the $m = 0$ and $m = N/2$ points are treated differently but, once we've programmed a function[24] to take care of these special points, we won't have to worry about them again. In many cases, the zero-frequency ($m = 0$; "DC") point and the point at the Nyquist frequency ($m = N/2$ or $f = f_s/2$) aren't important but you should still perform the calculation correctly. By being careful with the end points we've guaranteed that the single-sided spectral density also obeys a simple form of Parseval's Theorem:

$$\frac{1}{N}\sum_{n=0}^{N-1} x_n^2 \quad = \quad \sum_{m=0}^{N/2} G_{XX} \cdot \Delta f \quad . \tag{I-40}$$

[24] You should build your own library of functions for the basic operations described in this course. You will learn far more about the operations and you will be able to adapt your functions to your own applications. Many software packages have built-in functions but it is all too easy to make mistakes with these built-in functions as they often incorporate assumptions that may challenging to find and understand.

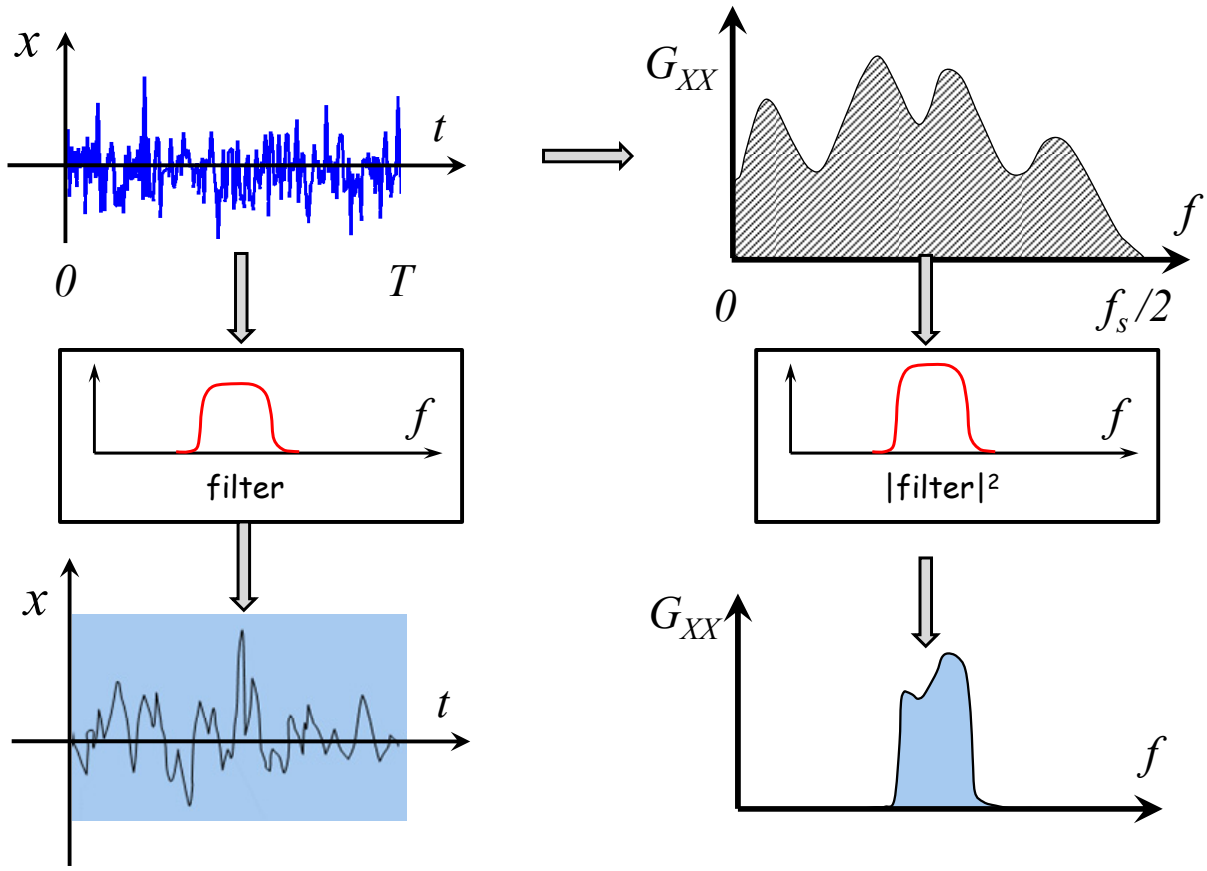


mean-square value of x_n = integral of G_{XX} from 0 to $f_s/2$

Having this function, G_{XX}, we can determine the true power in any portion of the spectrum. For human exposure or annoyance studies, for example, we may need to know the actual power in "octave" bands or "one-third-octave" bands or we may need to know the "A-weighted" level. All of these operations involve measuring the "power" in a particular band of frequencies. We can get that information and more from the power spectral density[25] (PSD), G_{XX}. For any filter function, we can use the spectral density to find the power from that filter by multiplying G_{XX} by the magnitude-squared of the filter's frequency response[26] and then integrating (sum times Δf). This will give the same answer as computing the mean-square in the time domain after the filtering operation.

[25] In these notes, I will often drop the word "power" and only infrequently use the notation PSD. Although common terminology, the word "power" is unnecessary and can be misleading. We will rarely work with the spectral density of power (in watts/Hz). Most often, we will work with the pressure or voltage spectral density in Pa^2/Hz or V^2/Hz. "Power-like spectral density" would be awkward.

[26] To be clear, we can multiply the complex linear spectrum by the complex frequency response and then construct the spectral density. This is equivalent to multiplying the spectral density by the magnitude-squared of the complex frequency response.



Exercise 1.3: Spectral Density

Generate an 8-point vector with a "random" selection of numbers – just type in some arbitrary sequence of integers (as you did in a previous exercise). Call that vector the time series. Since you are making up the time series, pick some Δt. That sets the sample rate, f_s, and the number of points and Δt set the duration of the time record, which, in turn, sets the frequency-domain, Δf. Find the linear spectrum, X_m, using the *fft* function. Find S_{XX}. Add all 8 values of S_{XX} and multiply the result by Δf. Compare that value to the mean-square value of the original time series. Find G_{XX}. How many values of G_{XX} are there? Add all of the values of G_{XX} and multiply by Δf. The result should equal the mean-square value of the original time series.

This check of the "integral" of the spectral density over frequency compared to the mean-square value in the time domain is an extremely valuable tool for avoiding errors.

Units

If the units of the sampled time series are pascals (Pa), then the units of the linear spectrum,

$$X_m = \sum_{n=0}^{N-1} x_n \, e^{-j2\pi\frac{nm}{N}} \, \Delta t \quad , \tag{I-41}$$

would be pascal·seconds (or, equivalently, Pa/Hz). This would make the units of the spectral density,

$$G_{XX} = \frac{2}{T}\left|X_m\right|^2 \quad , \tag{I-42}$$

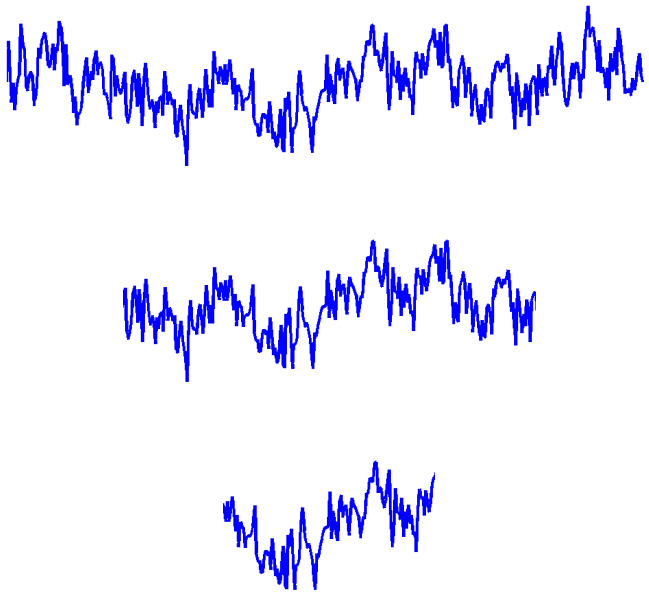
Pa^2s^2 over seconds or Pa^2s or, equivalently, Pa^2/Hz. These units, Pa^2/Hz, are appropriate for the description of a power-like quantity (i.e., pascals-squared) distributed over frequency.

You will often see displays of the square root of the spectral density. This leads to the awkward unit of pascals per root hertz (Pa/√Hz). Don't let these units confuse you. When you see a "per root hertz" unit, remember that it was the result of taking the square root of a distribution of a power-like quantity over frequency. The power-per-hertz becomes amplitude-per-root-hertz when you take the square root.

Energy spectral density

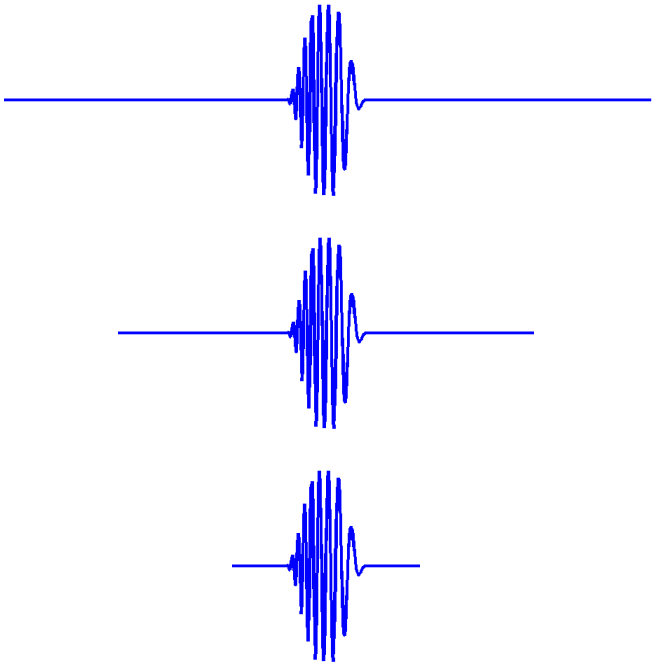
In the discussion above, we have taken the view that we are describing power and power-like quantities. This version of spectral density is often called power spectral density (PSD) even though we may not be working with units of power. For more-or-less continuous signals, the description in power or power-

like quantities is appropriate. For such signals—stationary[27] signals—the results depend very little on the length of time (the "record" length) we examine. For example, we would obtain nearly the same mean-square value using different length records:



For a time-limited signal (an explosion or a short pulse, for example), the mean-square value would depend on the length of the record because the averaging process is taken over the entire record regardless of whether there is real signal energy over the entire record. The mean-square values for the following three records would be different even though the actual signal is identical in all cases:

[27] I will not give a precise definition for stationarity. In dealing with real signals, a crude, intuitive "definition" serves well. For our purposes, a stationary signal is one for which simple measures (the mean, the mean-square, the distribution of power with frequency) have sufficiently small dependence on time that we can ignore the time dependence without introducing unacceptable error. This "definition" is a far cry from the definitions of stationarity offered in classical stochastic-theory texts but the classical degree of stationarity is rarely (if ever) achieved in real signals. Our definition is an operational one.



For short-duration signals, a description in energy terms is more natural. The energy spectral density (ESD) would be the same for all three cases shown above. ESD is defined in much the same way as power spectral density: the difference is that the quantity of interest is not averaged, it is simply integrated. The time/frequency relationship for ESD is,

$$\sum_{n=0}^{N-1} x_n^2 \cdot \Delta t \quad = \quad \sum_{m=0}^{N-1} ESD \cdot \Delta f \quad . \tag{I-43}$$

There is a difference of a factor of T (the record length in seconds) between the power spectral density, G_{XX}, and the energy spectral density, ESD. The left side in the equation above is the integral of the square of the time-series values; the right side is the integral of the energy spectral density. These quantities are proportional to energy, not power. The double-sided ESD is the square of the magnitude of the linear spectrum,

$$ESD \quad = \quad \left| X_m \right|^2 \quad . \tag{I-44}$$

(The more convenient single-sided energy spectral density would be constructed as the single-sided power spectral density was: double the values of the positive-frequency components except for the component at zero frequency and the component at $f_s/2$, then discard all of the negative-frequency components. An alternate approach: construct G_{XX} and then multiply G_{XX} by the record length, T.)

For a time series in acoustic pressure, the units of power spectral density are Pa^2/Hz; the units of the corresponding energy spectral density would be Pa^2 s/Hz (or, equivalently, Pa^2 s^2 or Pa^2/Hz^2). Of course energy spectral density can be used for true energy units, in which case the units would be simply joules per hertz.

Power spectrum

Another quantity in common use is the power spectrum. Unfortunately, this term is sometimes used as a synonym for spectral density. In its proper use, power spectrum is spectral density times Δf. For a pressure basis, the units would be Pa^2. Power spectrum is often used when the signals of interest are sine waves. With no time-domain windowing (see discussion of windows in the next chapter) and a sine wave at a frequency equal to one of the spectral bin centers, the mean-square value of the sine wave is equal to the value of the power spectrum at the sine-wave frequency. We will rarely use power spectrum in these notes as spectral density is a more flexible quantity; spectral density permits expression of both sinusoidal and broadband signals with equal ease and accommodates windowing and off-center sine waves naturally. (Time-domain windowing is accommodated in the power spectrum by multiplying the spectral density by the "effective noise bandwidth" or ENBW instead of by Δf: see the discussion of ENBW in the next chapter.)

Exercise 1.4: Spectral Density of a Sine Wave

Generate a sine wave – your choice of amplitude, frequency, and sampling rate. Find Δf. Make the frequency an integer multiple of Δf (so that there is no discontinuity in the implied periodic extension above and below your record of N points).

Find G_{XX} for this sine wave. One value of m should give a much higher G_{XX} value than any other. The units of G_{XX} are amplitude-squared per hertz. Calculate Δf for your choice of sampling rate and length of record. Then find the root-mean-square (*rms*) value of the sine wave from the single peak value of G_{XX}:

$$rms = \sqrt{\left(G_{XX}\right)_m \cdot \Delta f} .$$

This value should equal the amplitude of your original sine wave divided by the square root of two.

Create a vector with the frequencies that correspond to each value of G_{XX}. Plot G_{XX} as a function of frequency. Does the peak appear at the correct frequency? If it doesn't, then you've done something wrong! Develop the habit of performing simple checks like this.

Try changing the frequency of the sine wave by a fraction of Δf. There will now be several values that make up the peak in G_{XX}. Find the *rms* value by "integrating" over these values.

Exercise 1.5: Aliasing and Under-sampling

You should do the previous exercise before attempting this one. For a sine wave with any frequency less than $f_s/2$, you should be able to get the correct mean-square value of the time-domain waveform from the "integral" of G_{XX}.

Try a sine wave with a frequency that is greater than $f_s/2$ (but NOT an integer multiple of f_s). Plot the sampled waveform in the time domain. This sine wave is under-sampled (try a frequency that is just a few Δf less than f_s). The peak in the spectral density will appear at the wrong frequency yet you will still get an accurate measure of its amplitude by "integrating" G_{XX}. This sine wave is "aliased" to a different part of the spectrum. (Aliasing isn't always bad. Sometimes it's used intentionally to translate a higher frequency but band-limited signal down to a lower frequency.)

Can you deduce the "rule" for the frequency at which a sine wave with a frequency greater than $f_s/2$ appears in G_{XX}? Why did I caution you not to use an integer multiple of f_s for the frequency? Try it and look at the sampled time series.

Changing the sample rate

There are occasions when it can be useful to change the sampling rate of a time series. For example, you might have one recording sampled at 44 100 samples per second (the standard "compact disk" sample rate) and another recording sampled at 48 000 samples per second (S/s or Hz). To find the time shift of a common signal between the two recordings, you could calculate the cross-correlation (to be described in a later chapter); however, making this calculation is difficult if the sample rates are not equal. If you resample the 44 100 S/s recording to 48 000 S/s, then subsequent processing is straightforward.

With resampling, you can even compensate for small clock differences in acquisition systems. If you are generating a signal using a digital-to-analog converter and then receiving that signal using a separate analog-to-digital converter, the normal (small) clock-rate difference between the two converters may hinder your analysis. Matching the timing of these two channels of data may only require very small adjustments to the sample rate.

Furthermore, one aspect of proper sampling that is often not appreciated is the visual impact of a sampled signal. While the sampling theorem (to preserve the original signal content in a sampled version of that signal, the sample rate must be more than twice the frequency of the highest frequency component in the original signal) suggests, for example, that sampling a 200 Hz sine wave at 410 S/s (or Hz) preserves all of the information in that sine wave, plotting the sampled points gives a picture that looks very little like a sine wave. If, however, you take the sampled version at 410 S/s and increase the sample rate to, say, 10 000 S/s, the new time series will plot as a visually pleasing sine wave even though no new information was added. This process is often called zero-padding but is described below as the fourth method of changing the sample rate.

There are resampling functions in software packages (like *resample* in MatLab) but they are often restricted to integer factors: changing a sample rate from 24 000 S/s to 48 000 S/s is a simple factor of two; changing from 44 100 to

48 000 would require increasing the 44 100 rate[28] by a factor of 160 and then decreasing that new rate by a factor of 147. (After reading the rest of this section, you might consider why the "equivalent" process of reducing the 44 100 rate by a factor of 147, then increasing the resulting rate by a factor of 160 is a horrible idea and not at all equivalent.) This approach to resampling is inefficient even for these two rates that share a large common factor (300). Furthermore, integer resampling must be tailored to the specific sample rates.

For reducing the sampling rate by an integer factor, the MatLab *decimate* function can be used. This function combines anti-alias filtering and down-sampling. The filter can create problems if the down-sampling factor is too large (if you need to down-sample by a factor larger than 10, do the down-sampling in several stages).

Another approach is interpolation. Analysis packages also often have functions for interpolation. The *interp1* function in MatLab will interpolate values using a vector as the basis. These interpolation functions would generate poor results, though, for large changes in rate. They should also only be used for increasing the sample rate (up-sampling) as there is no explicit provision for preventing aliasing in down-sampling and for small changes in sampling rate. If the desired change in sample rate is more than a factor of two, the next technique may be more effective.

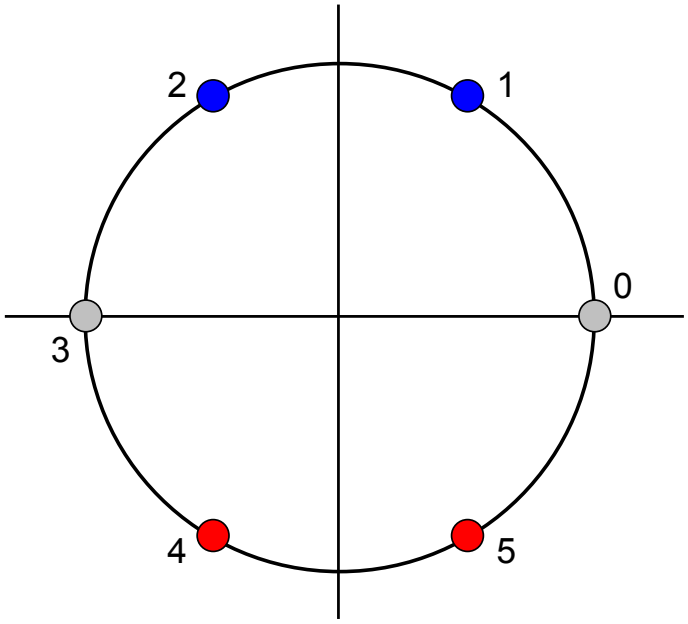
A fourth approach can be effective in many circumstances. This approach takes advantage of the band-limited nature of the already-sampled signal and, for up-sampling, the process is reversible. This technique is equivalent to exact Fourier interpolation.

Up-sampling

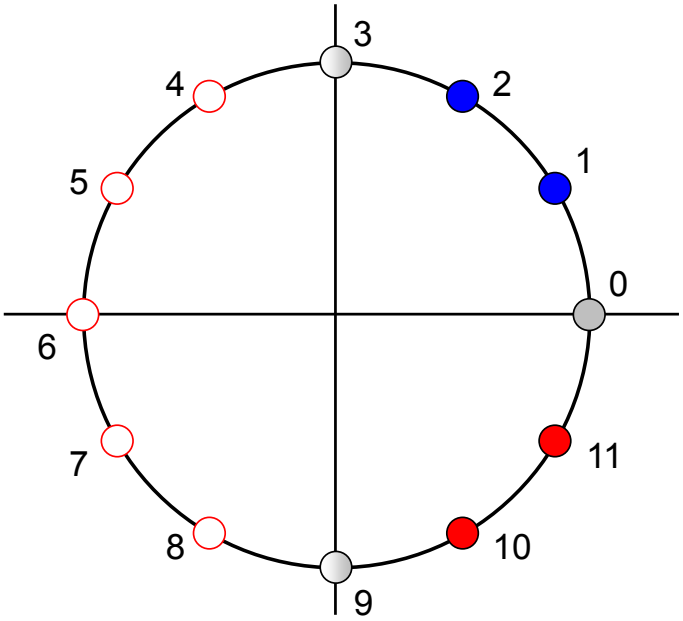
Take the original time series (N points long with a sample rate of f_s) and find the linear spectrum. To understand the Fourier up-sampling process, consider the

[28] 44 100 equals $(2*3*5*7)^2$. 48 000 equals $3*2^7*5^3$. Both of these numbers are highly factorable, a property that modern FFT routines exploit for increased speed. FFT sizes that are integer factors of two allow the most optimization in an FFT but good algorithms make use of other factorizations to improve efficiency.

unit-circle view of points in the linear spectrum (remember, the unit-circle view just shows the complex values of the complex exponential, not the actual linear-spectrum values). If N is originally six, the six exponentials would distribute evenly around the unit-circle:



To increase the apparent sampling rate, we add zero-value points to frequencies above the original Nyquist rate. To do this, take the linear spectrum value at the original $f_s/2$ frequency (point 3 in this example) and split it into two points, each with half the original value. Then add enough zeros in between these two half-value points to bring the total number of points to the new number, N_1. In the diagram below, we've brought the total number of points up to 12; points 3 and 9 are the half-value points produced by splitting the original $f_s/2$ point.



The new set of 12 points is now re-distributed evenly around the unit circle. (The unit circle is a visual aid. You don't have to take any steps to re-distribute the points; the re-distribution is automatic.) Now take the inverse transform. The resulting time series will have twice the original sampling rate. If you need to recover the original time series, reverse the steps.

Since the duration, T, of the record has not changed, the sample rate, which equals N_1/T, has changed. In this example, N_1 is twice N so the new sample rate is twice the original sample rate. While the example is for an integer factor increase in sample rate, the process is not restricted to integer rate-factor changes. For the desired new rate, find the value of N_1 that corresponds to the new rate (N_1/T) and add the corresponding number of zeros to the linear spectrum.

If you are not careful with the Δt factors in the forward and inverse transforms, your up-sampled signal will have a different amplitude than the original signal. Use the original Δt to find the linear spectrum. After adding zeros, you'll use the inverse transform to reconstruct the time series but be sure to use the new Δt in that inverse transform. In summary,

Find the linear spectrum at the original sample rate, f_{s0}:

$$X_m \quad = \quad fft\left(x_n\right)\Delta t_0 \quad \text{where} \quad \Delta t_0 \;=\; 1/f_{s0} \quad , \tag{I-45}$$

insert the extra zeros (which changes the sample rate), then find the inverse transform using the new sample rate, f_{s1}:

$$x_n \quad = \quad ifft\left(X_m\right)/\Delta t_1 \quad \text{where} \quad \Delta t_1 \;=\; 1/f_{s1} \quad . \tag{I-46}$$

Down-sampling

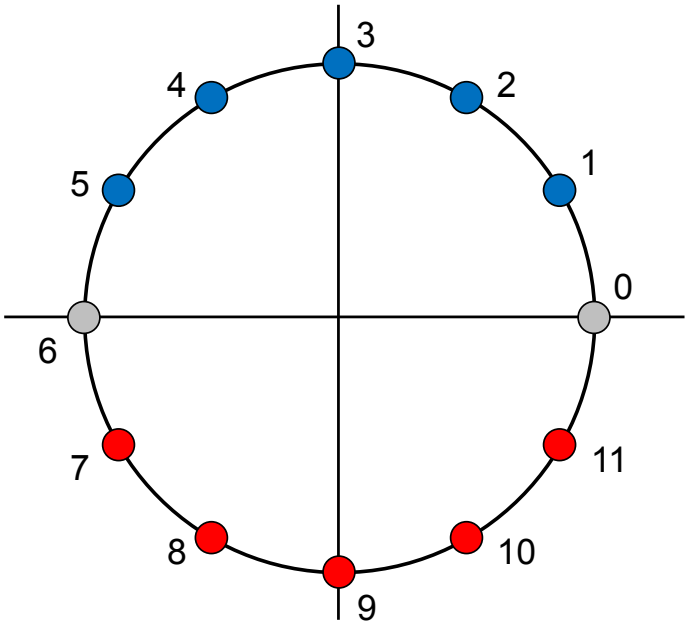
There is a critical difference between up-sampling and down-sampling. In down-sampling, the sample rate is reduced so the range of frequencies that can be represented properly is also reduced. Any energy outside of the new, small frequency range will be aliased to other frequencies in the process. Therefore, any energy originally in this excluded frequency range (between the old and the new Nyquist frequencies) must be removed before down-sampling (and that energy is lost—down-sampling is not reversible).

If your objective is to match two sample rates, it is normally best to up-sample the lower rate rather than down-sample the higher rate so that you don't have to contend with removing signal energy. On the other hand, if your objective is to reduce the sample rate then you have no choice.

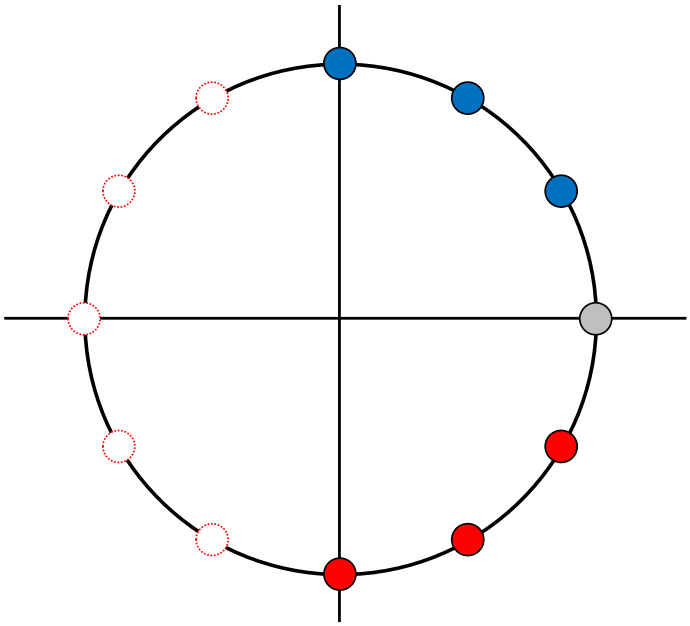
Filtering will be discussed in a subsequent chapter; for now, accept that filters can be constructed to reject some range of frequency. Filtering can be implemented in either the time domain or in the frequency domain. If you have a suitable time-domain filter, filter the time series first and then find the linear spectrum. If you prefer to filter in the frequency domain, find the linear spectrum first and then taper the energy that would be aliased down to negligible levels.

The unit-circle diagram below shows the arrangement of complex-exponential values for a transform with $N = 12$. If the goal is to reduce the sample rate by a factor of two, then (imagining the reverse of the up-sample process) we

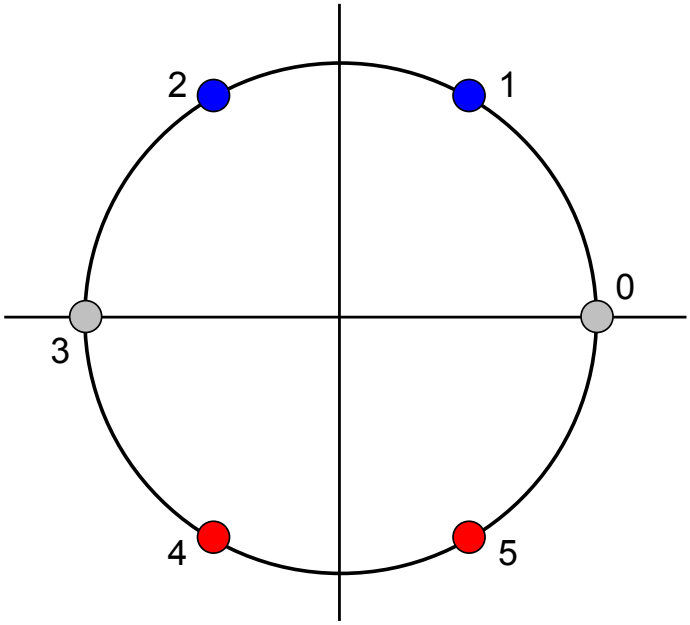
need to remove points 4 through 8 and sum points 3 and 9 to create the new $f_s/2$ value.



The next figure shows the points to be removed as dotted red circles with no fill color for the case of reducing the sample rate by a factor of two. While the anti-alias filtering could, in principle, be done simply by discarding the points to the left of the vertical axis, the abrupt change in linear-spectrum value at points 3 and 9 (for this example) could introduce non-physical artifacts into the resultant time series. If the time series has already been low-pass filtered (i.e., passing only the energy below the new Nyquist frequency), then the points to the left of the vertical axis can be simply discarded. A frequency-domain filter would smoothly taper the higher-frequency values of the linear spectrum to minimize problems in the time domain. With only 12 points, there aren't many points available for tapering but, if there were 12 000 points, then the hundred or so points nearest the vertical axis (and to the *right*) could be tapered gently down to zero.



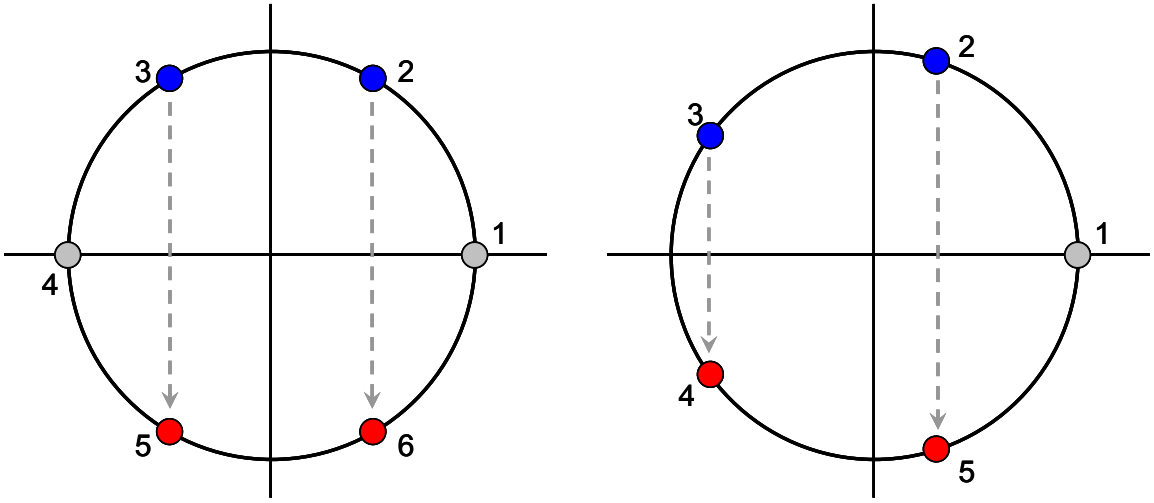
Once the filtering has been done, the extra points are removed to bring the total number of points down to the number corresponding to the new (lower) sample rate. In this example, the original points 4 through 8 are discarded and 3 and 9 are summed to become the new point 3 in the diagram below:



Taking the inverse transform of this smaller linear spectrum produces the down-sampled time series. Make sure you use the proper Δt factors in the forward and inverse transforms (see Up-sampling section) to preserve signal amplitude.

Handling an odd number of points

In this introduction to the fundamentals of time series and spectra, we've assumed that the number of points in the time series is even. A consequence of this choice is that the points in the single-sided spectral density at $f = 0$ and at $f = f_s/2$ are treated differently than all of the other points. If, however, there are an odd number of points, then there is no point in the linear spectrum at exactly $f_s/2$. At times it is convenient to work with an odd number of points so it is useful to understand what needs to be done differently. The following pair of diagrams is useful for illustrating the difference. These diagrams represent the values that the complex exponential factor assumes in either the forward or the reverse discrete Fourier transform.



The circles represent the end points of the complex exponential factors where the vertical coordinate represents the imaginary part and the horizontal coordinate represents the real part. The end points of the complex exponentials in the discrete Fourier transform (forward or reverse) are evenly distributed around the unit circle on a plot like this one. The figure on the left is for $N = 6$. If there are an even number of points in the transform, then one of the complex exponential factors will be exactly negative one (the grey circle labeled "4"). The horizontal axis is a line of complex-conjugate symmetry. Pairs of points on the same vertical line (e.g., points 2 and 6 in the diagram on the left) are complex conjugates and, if the linear

spectrum of a real time series has this same complex conjugate relationship: point 6 (a red circle) in the linear spectrum must be the complex conjugate of point 2 (the blue circle on the same vertical line). Points 1 and 4 (grey circles) do not have symmetry partners – these points must be real. Notice that point 4 is the $N/2 + 1$ point. This much is consistent with the treatment in this chapter for an even number of points.

For an odd number of points, there is no point at exactly negative one. Every point except the first point (the point that corresponds to $f = 0$) has a symmetry partner. In the example above (the right-hand diagram), $N = 5$, but there are only 3 unique points. Points 4 and 5 are complex conjugates of points 3 and 2, respectively. Points 2 through $(N + 1)/2$ are the "positive-frequency" points and, along with the $f = 0$ point, determine the linear spectrum of a real time series completely.

In summary, regardless of the number of points in a time series or its corresponding linear spectrum, the first point is the $f = 0$ point and it must be real if the time series is real. For an even number of points, the positive-frequency points in the linear spectrum are points 2 through $N/2 + 1$. If the time series is real and has an even number of points, the linear spectrum at $N/2 + 1$ must be real and the negative-frequency points ($N/2 + 2$ to N) must be the complex conjugates of the corresponding "mirror" points as in the diagram above. For an odd number of points, every point except the $f = 0$ point has a complex-conjugate partner and the positive-frequency points are the points 2 through $(N + 1)/2$. In other words, the single-sided spectral density for N even has $N/2 + 1$ points and the single-sided spectral density for N odd has $(N + 1)/2$ points.