Dynamic Signal Analysis Basics

Andrew Snyder
Crystal Instruments Corporation
4699 Old Ironsides Drive, Suite 100
Santa Clara, CA 95054, USA

Table of Contents

Basic Theory of Frequency Analysis

Dynamic Signal Analysis (DSA) is an application area of Digital Signal Processing (DSP) technology. Compared to general data acquisition and time domain analysis, DSA focuses more on the dynamic and frequency aspects of signals such as frequency response, dynamic range, total harmonic distortion, phase match, and amplitude flatness. In recent years, time-domain data acquisition devices and DSA instruments have converged. More and more time domain instruments, such as oscilloscopes, can do frequency analysis while more and more dynamic signal analyzers can do long time data recording.

Among the DSP applications used in Dynamic Signal Analysis, the most fundamental and popular is the Fast Fourier Transform (FFT) algorithm. The FFT transforms time domain signals into the frequency domain very efficiently. To use FFT-based measurements effectively, however, one must understand the fundamental issues and computations involved. This chapter describes some of the basic signal analysis computations, discusses the anti-aliasing and acquisition front end for FFT-based signal analysis, explains how to use windowing functions correctly, explains some spectrum computations, and shows FFT-based functions for some typical measurement examples.

The Fourier Transform

Jean Baptiste Joseph Fourier made his famous statement in the 19th century that all periodic signals can be represented as an infinite sum of sinusoidal functions. In other words, any periodic time domain signal $x(t)$with period T is equal to a sum of complex exponentials:

$$x(t) = a_0 + a_1 e^{-j\frac{2\pi}{T}t} + a_2 e^{-j_2\frac{2\pi}{T}t} + \cdots$$

This is called the **Fourier Series**. The complex exponentials $e^{-j\frac{n2\pi}{T}t}$ are the *frequency components* with amplitute values a_n. The frequencies are multiples of the **fundamental frequency** $\frac{1}{T}$. The (infinite) set of these values $\{a_0, a_1, a_2, \dots\}$

is the *frequency domain* representation of the *time domain* signal $x(t)$. No information is lost in going from one domain to the other; in fact the signal $x(t)$ can be perfectly reconstructed from the set of a_n.

This decomposition can be extended to non-periodic signals. As the period of the signal increases, the fundamental frequency decreases and the frequency components become more closely spaced. Taken to the infinite limit, this spacing becomes infinitesimal and the frequency domain representation becomes a continuous function $X(f)$. $X(f)$ is the **Continuous Time Fourier Transform (CTFT)** of $x(t)$, defined as

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

In general, the Fourier Transform is defined for all frequencies from negative to positive infinity and is complex-valued. However, for real-valued time signals the frequency transform must be conjugate-symmetric about the zero frequency and the negative frequencies are therefore redundant. It is common, therefore, to consider only the positive frequencies.

The Discrete Fourier Transform

The CTFT, in general, does not have an analytic solution and must be solved numerically. To do this, the infinite integral must be replaced by a finite sum.

The first step is to approximate the continuous time signal with a set of discrete values. This is the process of **sampling**, which takes 'snapshots' of the signal at evenly spaced points in time. This spacing is ΔT and its inverse, $\frac{1}{\Delta T}$, is the *sample rate* or number of sample points per second. This process generates a discrete form of the original time signal, $\hat{x}(n)$, with $\hat{x}(1) = x(t_1), \hat{x}(2) = x(t_2)$, etc., where $t_n = n\Delta T$.

According to Nyquist's theorem, the original signal can be perfectly reconstructed from this sampled version as long as it does not contain any frequency

components greater than half the sample rate. If components above this **Nyquist Limit** are present in $x(t)$, they will become *aliased* in $\hat{x}(n)$ and appear as new, lower-frequency components. If the sample rate is f_s, and a frequency f_1 is present in $x(t)$ with $f_1 > \frac{f_s}{2}$, then $\hat{x}(n)$ will contain the frequency $f_1 - \frac{f_s}{2}$. This is undesirable distortion in the sampled signal.

To prevent this, a low-pass **anti-aliasing filter** is used. Ideally, this filter would pass, with no loss, all frequencies up to but excluding $\frac{f_s}{2}$. Such a filter is impossible to realize in reality and, instead, the roll off occurs over finite bandwidth. Because of this, the maximum usable frequency in signal analyzers is actually less than half the sample rate.

With a discretized signal, the Fourier Transform now involves an infinite sum over infinite time. Most users aren't willing to wait this long for an analysis, so the signal must be broken into finite *blocks* of N samples. Each block is transformed with the **Discrete Fourier Transform (DFT)**:

$$\hat{X}(k) = \sum_{n=0}^{N-1} \hat{x}(n) e^{-jk\frac{2\pi}{N}n}$$

This transforms the discrete signal $\hat{x}(n)$ over a period of N samples into a discrete frequency domain representation $\hat{X}(k)$, where n and k are integers. The N time values become N complex frequency values.

$\hat{X}(k)$ is defined for all k, but only has N unique values. This is possible because $\hat{X}(k)$, is periodic, meaning the values between $-\frac{N}{2}$ and $\frac{N}{2}$ are repeated infinitely many times in the positive and negative direction. $\hat{X}\left(\frac{N}{2}\right)$ is the same as $\hat{X}\left(-\frac{N}{2}\right)$, and $\hat{X}(0)$ is the same as $\hat{X}(N)$. Because of this, the DFT is sometimes described as being defined from $-\frac{N}{2}$ to $\frac{N}{2} - 1$, but can also be defined from 0 to $N - 1$.

For real-valued time signals, $\hat{X}(k)$ is conjugate-symmetric. This means that the real part of $\hat{X}(k)$ is equal to the real part of $\hat{X}(-k)$, and the imaginary part of $\hat{X}(k)$ is equal to the *negative* of the imaginary part of $\hat{X}(-k)$. The spectrum, then, is evenly split between the negative and positive frequencies, and both sides contain identical information. This gives rise to the *one-sided DFT*, which is defined for $0 \leq k \leq \frac{N}{2}$.

Computing the DFT sum is computationally intensive, and computer scientists have developed a much more efficient algorithm called the **Fast Fourier Transform (FFT)**. The FFT transforms each block of N samples into $\frac{N}{2} + 1$ complex-valued positive frequency *lines*, equivalent to the one-sided DFT. The most common FFT implementation is the Radix-2 DIF FFT algorithm, which requires that the block size N be a power of 2 ($N = 2^m$, where m is an integer).

DFT Frequency Range and Resolution

The original Fourier Series decomposes an infinitely long time signal into infinitely long periodic functions. The DFT (and FFT, which are mathematically equivalent) attempts to decompose a *finite* discrete time signal of N points into *infinite* periodic functions. However, a finite function cannot be formed from infinite functions. The DFT actually treats the N points as one period of an infinite periodic signal.

In other words, the DFT assumes a fictitious time signal where the original N points are repeated over and over, ad infinitum. This new signal has a period equal to the time length of those original N points, or $\frac{N}{f_s}$. Its fundamental frequency $\frac{f_s}{N}$, which is the reciprocal of its period, is the lowest frequency present, and all other frequency components are multiples of this value. The highest frequency is the Nyquist frequency $\frac{f_s}{2}$, limited by the sample rate f_s.

In summary, the one-sided DFT takes N time samples and returns $\frac{N}{2}+1$frequency component amplitudes at $\left\{0, \frac{f_s}{N}, \frac{2f_s}{N}, \ldots, \frac{f_s}{2}\right\}$. The component at the zero frequency is just the arithmetic average of the time points.

The frequency range of the transform is $\frac{f_s}{2}$, determined by the sample rate in the time domain. The frequency resolution (spacing between the frequency lines) is $\frac{f_s}{N} = \frac{1}{T}$, determined by the time length of the block. The sample rate — the time *resolution* —affects the frequency range; the time length of the block — the time *range* —affects the frequency resolution. *Range* in one domain controls the *resolution* in the other domain.

Leakage

The DFT works very well if all the frequency components present in the original time signal are multiples of the fundamental frequency of the N-point blocks. In this case, an integer number of cycles fit perfectly within one block (Figure 1, right side) and the input assumed by the DFT is an accurate representation of the original time signal. The frequencies actually present in the original signal exactly match the frequencies of the DFT components.

In general, however, this won't be the case and the signal won't line up so nicely (Figure 1, left side). Each frequency component of the DFT no longer corresponds to a single, discrete frequency of the original signal, but instead represents a range of frequencies. This limits the resolution of the analysis and is a consequence of using only N points of the original time signal.

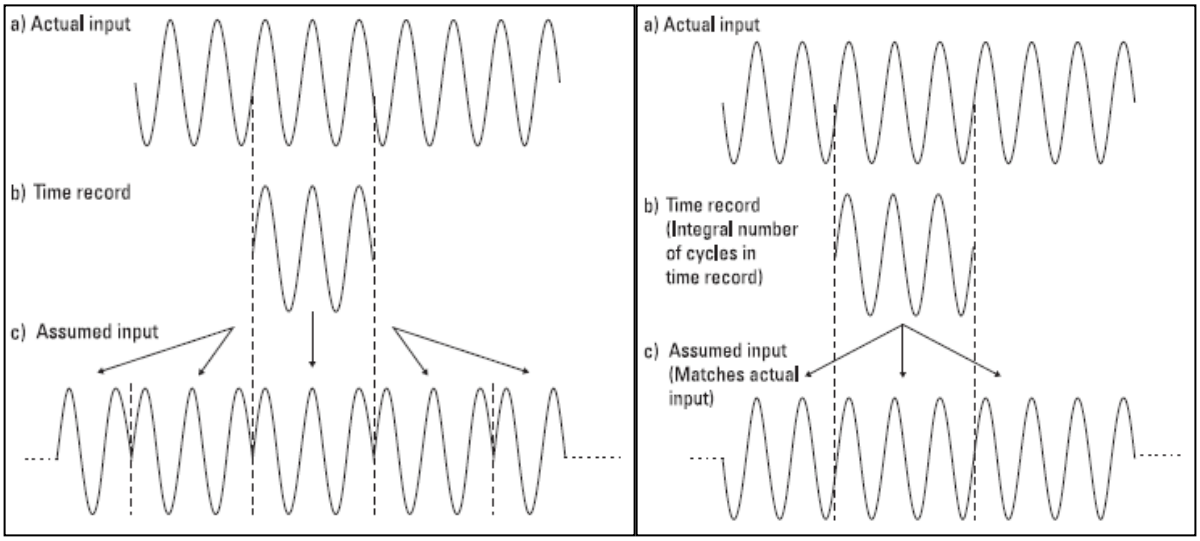

Figure 1. Relationship between actual input, block time record, and input assumed in FFT calculation

The components of the DFT correspond to frequency lines of the FFT, and each line has the frequency response shown in Figure 2.

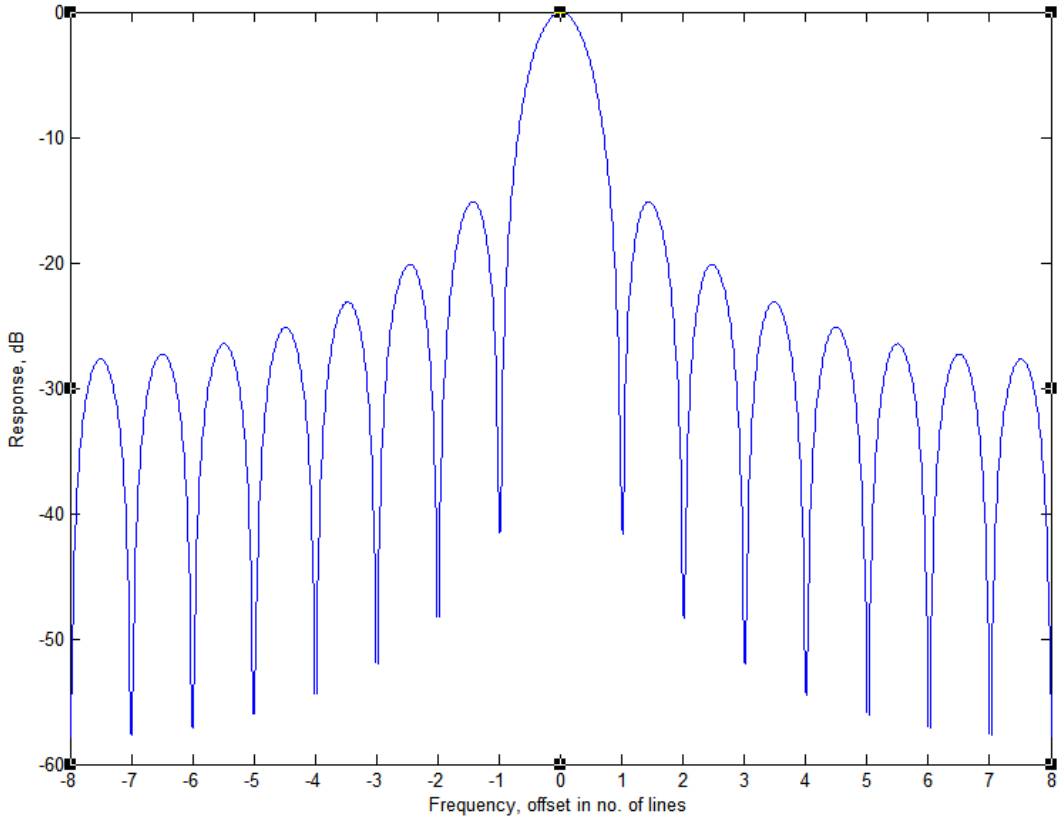

Figure 2 Response of FFT lines.

In this response shape, there is one main lobe and many side lobes that extend through the entire frequency range. The response is highest at the center frequency and is zero at the center frequency of every other line. In between the other lines it is not zero, and this results in inaccurate measurement of these frequencies. Figure 3. FFT measurement of sine wave, centered on frequency line on the left shows the FFT display when measuring a sine wave with a frequency exactly matching the center frequency of an FFT line. It is displayed perfectly. Figure 3, right side shows the display when measuring a sine wave of a frequency between two frequency lines. Here, even though there is only one frequency present it appears to have a much wider bandwidth. This effect is called *leakage* and can make it impossible to distinguish two frequency components that are close together.

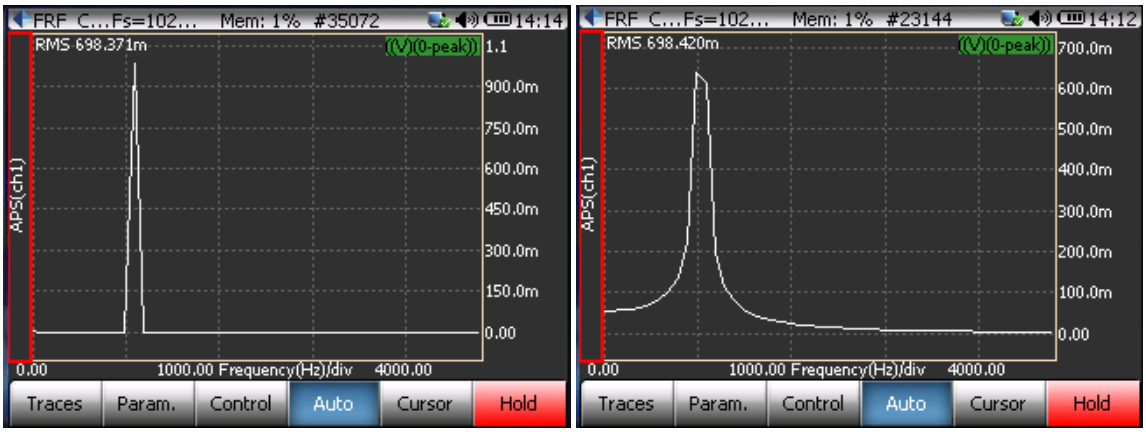

Figure 3. FFT measurement of sine wave, centered on frequency line (left) and not centered on a frequency line (right)

In general, leakage will always occur and will limit the resolution of any frequency display. There are two exceptions: one has been described above where the fundamental frequencies of the block and of the time stream match; the other is when the entire time signal completely fits within a single block. This occurs when measuring transient signals that are zero at the start and end of the block. These are often called **self-windowing functions**.

Windowing Functions and Scaling Factors

Leakage can be reduced, and measurement precision increased, by the use of **windowing functions**. These functions are defined over one block, are positive, and are zero at the beginning and end of the block. When the time signal blocks are multiplied by these functions before the FFT algorithm, the response shape of each frequency line is changed. With the correct function, this shape can be improved allowing for increased frequency resolution or accuracy. Various windowing functions have been developed, and each has a different effect on the response shape and are best suited for different applications. These functions are discussed in detail in a later section.

Because the windowing function attenuates a portion of the original data, it has a scaling effect on the overall amplitude of the frequency components. To correct for this and maintain accuracy, an **amplitude correction factor** C is used:

$$C = \sum_{n=0}^{N-1} w(n)$$

Amplitude and Spectrum

The amplitude values given by the DFT are complex, meaning they have real and imaginary parts. They can be converted to a magnitude and phase angle; however, in most applications, the phase angle of one signal by itself is meaningless.

The frequency component k of a signal block $x(n)$ is defined as

$$S_x(k) = \frac{2}{C} \sum_{n=0}^{N-1} w(n) e^{-\frac{j2\pi kn}{N}}$$

where $w(n)$ is the windowing function. n and k are integers, with $0 \leq n \leq N-1$ and $0 < k < \frac{N}{2} - 1$ (k is the frequency in *cycles per block*). C is the amplitude correction factor, described above. The factor of 2 is a correction required because this is a one-sided transform; a two-sided transform would have half the amplitude at the positive frequency and half at the negative frequency. There is an exception for $k = 0$ and $k = \frac{N}{2}$, which, in the two-sided spectrum do not have a corresponding negative frequency component. For these frequencies, therefore, the one-sided spectrum is defined without the factor of 2:[1]

[1] This is cumbersome, but is necessary, and forgetting these little details is a common source of bugs in DSP software.

$$S_x(k) = \frac{1}{C}\sum_{n=0}^{N-1} w(n)e^{-\frac{j2\pi kn}{N}}, k = 0 \text{ and } k = \frac{N}{2}$$

The real parts of S_x correspond to cosine components and the imaginary parts to sine components (90° phase difference). To convert to magnitude and phase, the magnitude is given by

$$\|S_x(k)\| = \sqrt{S_x^*(k) \cdot S_x(k)}$$

and the phase by

$$\angle S_x(k) = \tan^{-1}\frac{\Im(S_x)}{\Re(S_x)}$$

where $*$ denotes the complex conjugate, $\Re$ denotes the real part, and $\Im$ the imaginary part.

The magnitudes of the frequency components are collectively called the **Amplitude Spectrum**.

Power Spectrum

In many applications, the quantity of interest is power, or rate of energy transfer, which is proportional to the squared magnitude of the frequency components. The squared magnitudes of all the DFT frequency lines are collectively referred to as the **Power Spectrum**, defined as:

$$S_{xx}(k) = \frac{1}{2}S_x^*(k) \cdot S_x(k)$$

The power spectrum normalized to the width of the frequency lines is called the **Power Spectral Density.** This is the signal power per 1 Hz of bandwidth. Normalizing to the bandwidth makes the measurement it independent of the frequency resolution used in the analysis. It is the most common quantity used when measuring broad-spectrum random signals.

When measuring transient or short-duration signals, the quantity of interest is usually the total energy present in each frequency band. The total energy per 1Hz bandwidth is proportional to the **Energy Spectral Density**, equal to the power spectral density multiplied by the time duration of the block. Multiplying by time makes the measurement independent of the block time length.

Cross Spectrum

While the discussion up to this point has described the frequency analysis of one time signal, the **Cross Spectrum** characterizes the relationship between the spectra of two signals. For two signals x and y, with frequency components $S_x$and S_y, it is defined as:

$$S_{xy}(k) = S_x^*(k) \cdot S_y(k)$$

It is the *correlation* between the frequency components of two related signals. While the Power Spectrum is real-valued, the Cross Spectrum is complex. This means that it also describes the phase alignment of the two signals.

Frequency Response Function

An important application of Dynamic Signal Analysis is characterizing the input-output behavior of physical systems. This is the domain of **network analysis**. With linear systems, the output can be predicted from a known input if the **Frequency Response Function** of the system is known. The Frequency Response Function $H(f)$ relates the Fourier Transform of the input $X(f)$ to the Fourier Transform of the output $Y(f)$ by the simple equation

$$Y(f) = H(f)X(f)$$

The fact that $H(f)$ is independent of the input is what makes the system linear. The inputs and outputs can be force, acceleration, velocity, or any other physical quantity. With the DFT, the simplest definition of the frequency response

function is the ratio of the output spectrum to the input spectrum. However, the best way to calculate this that is resistant to noise uses the cross-power spectrum:

$$H(f) = \frac{S_{xy}(k)}{S_{xx}(k)}$$

Like the Cross-Power Spectrum, the Frequency Response Function is complex-valued and has both magnitude an phase. The magnitude is the ratio of the output to input amplitude of each frequency, and the phase is the phase change between the output and input.

When measuring the input-output behavior of a system, there is always noise present that obscures the output. An important measure is how much of the output is actually caused by the input. Another correlation measure called **Coherence** is defined as

$$C_{xy} = \frac{|S_{xy}|^2}{S_{xx}S_{yy}}$$

The coherence is between 0 and 1, with 1 meaning the output is perfectly explained by the input. A coherence of 0 means the output and input are uncorrelated.

The FFT in the Real World

The previous sections described the theory that makes frequency analysis of signals possible. This section describes the details of how frequency analysis, usually carried out by the FFT algorithm, is implemented for real signals.

More on Data Windows

Windowing functions, defined above, can help reduce leakage and increase the precision of the frequency measurement. These functions affect the response

shape of each frequency line. Figure 4 shows this shape for four common windowing functions.

This response always has a main lobe, reaching the highest point in the center, and many side lobes that taper off in each direction. Ideally, a windowing function would produce a narrow main lobe and non-existent side lobes, but this is not possible. There is a trade-off between the width of the main lobe and height of the side lobes. The functions that give low side-lobe response have a very wide main lobe and vice-versa. Because of this, different functions are better suited for different analysis situations.

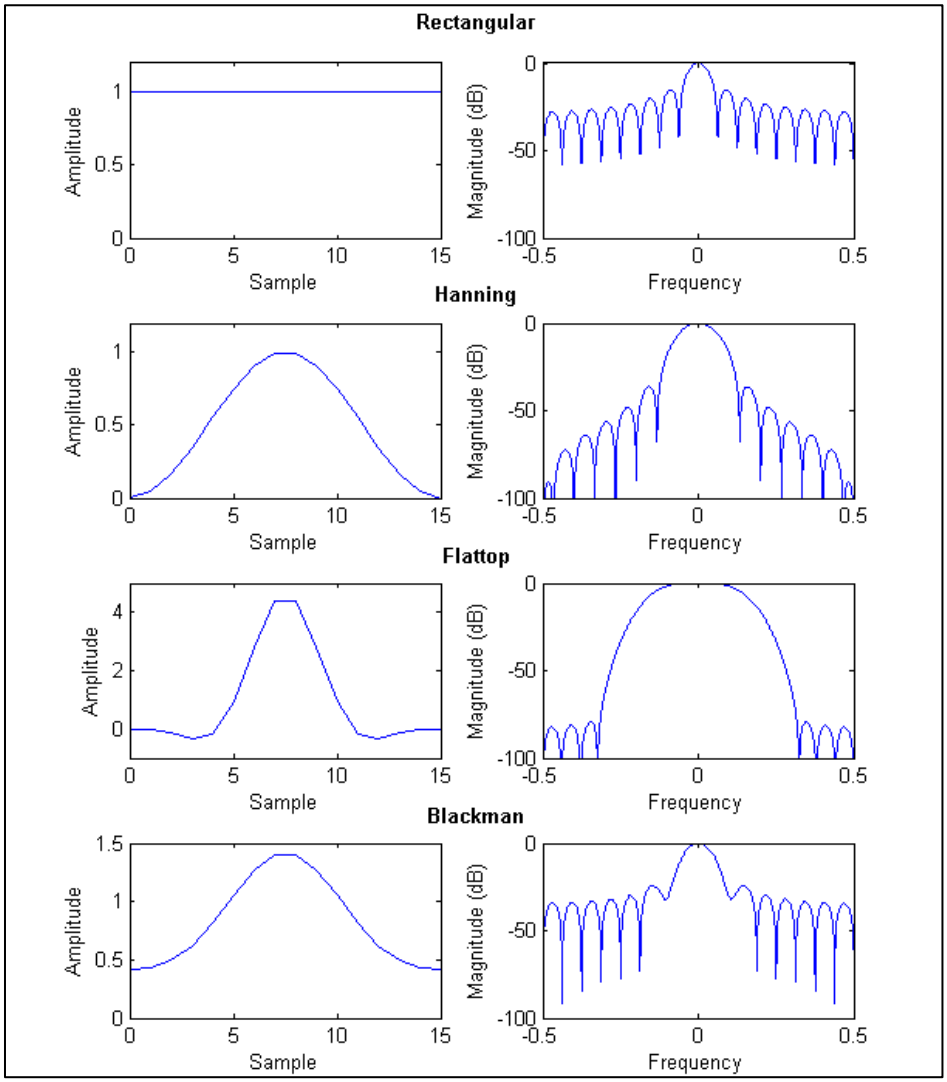

Figure 4. Spectral shape of common windowing functions.

Types of Windows

Each windowing function $w(k)$ is defined over one block, $-\frac{N}{2} < k \leq \frac{N}{2}$, with the block centered on $k = 0$. The equations of the most common functions are given below.

Uniform Window

$$w(k) = 1$$

The uniform or rectangular window is just unity over the entire block interval, and is equivalent to not using a windowing function at all.

Hann Window

$$w(k) = 0.5 - 0.5\ \cos(\frac{2\pi k}{N-1})$$

The Hann (or Hanning) windowing function is the most common because of its reasonable compromise between low side lobes and narrow main lobe. Figure 5 shows the FFT display for the same sine wave as **Error! Reference source not found.** but using the Hann windowing function.

Hamming Window

$$w(k) = 0.53836 - 0.46164\ \cos(\frac{2\pi k}{N-1})$$

The Hann and Hamming windowing functions are in the family known as "raised cosine" windows and are named after Julius von Hann and Richard Hamming respectively.

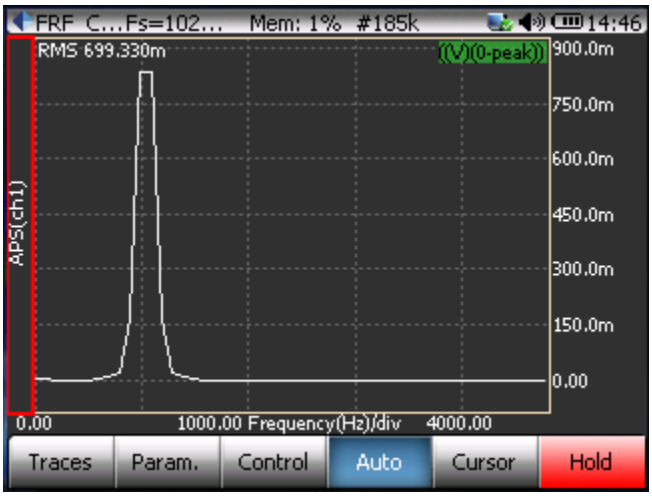

Figure 5. FFT measurement of sine wave with Hann window.

Blackman Window

$$w(k) = 0.84 - 0.5\cos\frac{2\pi k}{N-1} + 0.08\cos\frac{4\pi k}{N-1} \quad \text{for } k \in [0, N-1]$$

Flattop Window

$$w(k) = 1 - 1.93\cos\frac{2\pi k}{N-1} + 1.29\cos\frac{4\pi k}{N-1} - 0.388\cos\frac{6\pi k}{N-1} + 0.032\cos\frac{8\pi k}{N-1}$$

Kaiser Bessel Window

$$w(k) = 1.0 - 1.24\cos\frac{2\pi k}{N-1} + 0.244\cos\frac{4\pi k}{N-1} + 0.00305\cos\frac{6\pi k}{N-1}$$

Force Window

The force window is unity over some proportion of the block and zero over the rest. It is used when only part of the block contains useful information, as when measuring an excitation force with duration much shorter than the block length.

Exponential Window

$$w(k) = e^{\left(\frac{k\ln(R)}{N-1}\right)}$$

The shape of the exponential window is that of a decaying exponential with a final value of R.

Window Characteristics

Figure 4 shows four common windowing functions and the resulting response shape of the frequency lines.

The width of the main lobe is often described by the **half-power bandwidth**, which is the distance between the points where the response is -3 dB below the center peak. Similarly, the quarter-power bandwidth uses the -6 dB points. The width is given here as number of frequency lines, where each line is separated by a frequency of $\frac{f_s}{N}$.

The ability to distinguish two closely spaced frequency components increases as the main lobe narrows. However, as the main lobe narrows the response of the side lobes increase. There is a trade-off between amplitude accuracy and spectral resolution.

Side lobes occur on each side of the main lobe and are zero at the centers of other frequency lines. They are characterized by their peak response levels and roll-off rate. The side lobes affect the extent to which adjacent frequency components leak into the frequency lines. The side lobe response of a strong component can overpower the main lobe response of a nearby weak component.

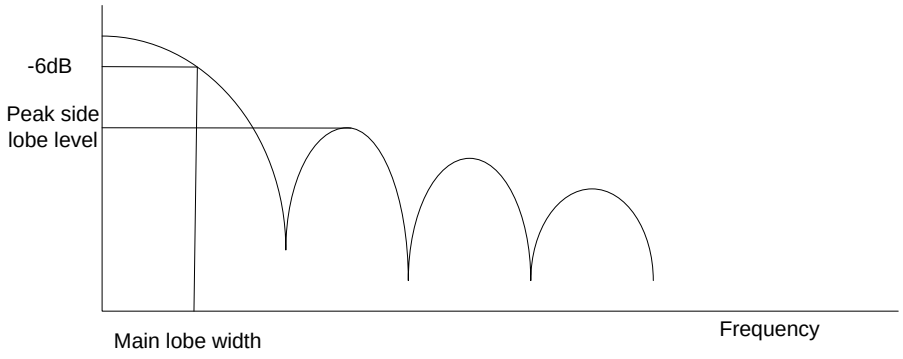

Figure 6. Window frequency response showing main lobe and side lobes.

Window	–3 dB Main Lobe Width (lines)	–6 dB Main Lobe Width (lines)	Maximum Side Lobe Level (dB)
Uniform (none)	0.9	1.2	−13
Hanning	1.4	2.0	−32
Hamming	1.3	1.8	−43
Blackman	1.6	2.3	−58
Flattop	2.9	3.6	−44

Choosing the Right Data Window

When measuring self-windowed functions, no windowing is required (use the Uniform window). When measuring period signals, it is best if the block time length is equal to the period of the signals.

If the goal of the analysis is to discriminate two or more sinusoids close together in the frequency domain, spectral resolution is very critical. For these applications, choose a data window with a narrow main lobe such as the Hanning window.

Use the Flattop window when amplitude accuracy is critical. The function generates a wide flat main lobe that sacrifices frequency resolution for accurate representation of amplitude.

For analyzing transient events such as impact and response signals, it is better not to use the windowing functions because they attenuate important information at the beginning of the sample block. Instead, use the Force and Exponential windows. A Force window is useful in analyzing shock stimuli because it removes stray signals at the end of the signal. The Exponential window is useful for analyzing transient response signals because it damps the end of the signal, ensuring that the signal fully decays by the end of the sample block.

Amplitude Units and Scaling

$S_x(k)$ can be thought of as a *vector* with both a magnitude and phase angle. When expressed as a magnitude, it can be scaled as a peak value, RMS value, or peak-to-

peak value. When expressed as a squared magnitude, it can be scaled to represent the power spectrum, the power spectral density, or energy spectral density.

Recall that $S_x(k)$ given as magnitude values is the Amplitude Spectrum. With no scaling, it represents the **peak** amplitude values of the sinusoidal functions that make up the signal. The **RMS**, or Root-Mean-Square, values represent the level that a constant (or DC) signal would need to have to transfer the same power as the component sinusoid. It is the square root of the average square value, equal to $\sqrt{2}$ times the peak value. The **peak-to-peak** value is the difference between the highest and lowest values of the sinusoid, equal to twice the peak.

In many measurement situations, the physical phenomena being observed involve transfers of energy. The rate of this energy transfer, which is defined as power, is usually proportional to the *square* of the measured quantities, such as voltage or velocity. Thus the squared magnitude values of the DFT, or the Power Spectrum, represents the signal power present in each frequency line. The section above on Power Spectrum describes the various ways it is scaled.

Averaging

In the steps described above, a sampled time signal was truncated into a block N samples long and multiplied with a windowing function. Then, the FFT algorithm was applied to the block resulting in N/2 complex-valued frequency lines.

In a typical analysis scenario, a time stream is broken into a series of blocks and each undergoes the FFT process. The results from each block are then combined in some sort of averaging function, giving an overall picture of the signal in the frequency domain. An identical result is obtained if the blocks are averaged in the time domain, before the FFT is applied. Since both the FFT and the averaging functions are linear transforms, their order does not matter.

Linear Spectrum Averaging

If the frequency amplitude values from each block are kept in their complex form and averaged, the result is a **Linear Spectrum Average** (also known as a **Vector Average**). The phase information in each block is preserved and frequency components in phase across blocks are reinforced while out of phase components are cancelled.

The primary utility of Linear Averaging is for **Time Synchronous Averaging**. This is used when the blocks can be aligned with the period of a fixed signal known to be present in the time stream. The alignment of the blocks makes the signal have the same phase in every block, while the phase of noise varies randomly. As more blocks are averaged together, the noise is cancelled out and the in-phase signal is reinforced.

Aligning the blocks in such a way requires a *synchronous trigger* to mark the period of the signal. Fortunately, this is usually available in measurement situations; for example, with rotating machinery a tachometer pulse that occurs once per revolution works very well.

Power Spectrum Averaging

If the power spectra of each block, which is in magnitude-squared rather than complex form, is averaged the result is a **Power Spectrum Average**. This destroys phase information, but often the phase isn't meaningful anyway. The power spectrum average gives an estimate of the average power distribution across the frequency range.

Fixed-Size Average, Exponential Average, Moving Average

There are different ways to mathematically combine data from many blocks into an average, whether this data is complex frequency amplitudes (linear spectrum average) or power spectra (power spectrum average).

A **fixed-size average** (sometimes confusingly called linear averaging, which has nothing to do with linear spectrum averaging described above) takes a specified number of blocks M and combines the numeric values in a regular arithmetic average:

$$G(k) = \frac{\sum_{m=1}^{M} S^m(k)}{M}$$

where $G(k)$ is the average for frequency k and $S^m(k)$is the frequency data for block m (either complex amplitudes S_x or power spectrum S_{xx}).

With fixed-size averaging there is one average every M blocks. However, with a continuous time stream, it is preferable to have an average that is updated every block. This can be done with a **moving average**, where an arithmetic average is calculated for the past M blocks every time a new block is acquired.

A moving or fixed-size average required enough memory to hold M blocks of data. A more efficient way to keep a running average is the **exponential average**. Here, only the current average is stored, and then updated when a new block is available by the equation:

$$G^{m+1}(k) = \alpha S^m(k) + (1-\alpha)G^m(k)$$

$G^m(k)$ is the previous average which is updated to the new average $G^{m+1}(k)$ with new spectrum data $S^m(k)$. $1/\alpha$ is the **average number**.

Overlap Processing

In the methods described above, one block is formed for every N samples of time data, which has a time duration $T = \frac{N}{f_s}$. For a large block size or low sample rate, this can be a considerable amount of time which causes the frequency data to update very slowly.

To decrease the time between updates, overlap processing can be used. While the block size remains N, a new block can be formed every D samples, with $D < N$. This means that, for each block, $D - N$ samples from the previous block were re-used in the new block. Often, this overlap is expressed as a percent, so an overlap ratio of 25% would mean that $D - N$ is 25% of N.

Overlap processing can form more signal blocks within a fixed total number of samples. When characterizing random signals, the accuracy of the frequency data depends, in general, on the total number of samples that go into the average and not the number of blocks. This means that, with a rectangular window, overlap processing will not give more accurate data for the same number of samples. However, with non-rectangular windowing functions some of the time domain data is attenuated, and overlapping blocks can help recover this. Overlapping can improve accuracy up until around a ratio of 50%.

References

To understand the topics of this article, we found the following three books to be very helpful:

1. Julius S. Bendat and Allan G. Piersol, **Random Data, Analysis and Measurement Procedures**, 2nd Edition, Wiley-Interscience, New York, 1986.

2. Julius S. Bendat and Allan G. Piesol, **Engineering Applications of Correlation and Spectral Analysis**, 2nd Edition, Wiley-Interscience, New York, 1993.

3. Sanjit K. Mitra and James F. Kaiser, Ed. **Handbook for Digital Signal Processing**, Wiley-Interscience, New York, 1993.