Fundamentals of Signal Analysis Series

Understanding Dynamic Signal Analysis

Application Note 1405-2

Agilent Technologies

Table of Contents

Introduction

This application note is a primer for those who are unfamiliar with the class of analyzers we call dynamic signal analyzers. These instruments are particularly appropriate for the analysis of signals in the range of a few millihertz to about a hundred kilohertz.

In this note, we avoid using rigorous mathematics and instead depend on heuristic arguments. We have found in over a decade of teaching this material that such arguments lead to a better understanding of the basic processes involved in dynamic signal analysis. Equally important, this heuristic instruction leads to better instrument operators who can intelligently use these analyzers to solve complicated measurement problems with accuracy and ease.*

In Application Note 1405-1, Introduction to Time, Frequency and Modal Domains, we introduced the concepts of the time, frequency and modal domains and the types of instrumentation available for making measurements in each domain. We discussed the advantages and disadvantages of each generic instrument type and noted that the dynamic signal analyzer has the speed advantages of parallel-filter analyzers without their low-resolution limitations. In addition, it is the only type of analyzer that works in all three domains.

In this application note, we will develop a fuller understanding of dynamic signal analyzers. We begin by presenting the properties of the Fast Fourier Transform (FFT), upon which dynamic signal analyzers are based. We then show how these FFT properties cause some undesirable characteristics in spectrum analysis like aliasing and leakage. Having demonstrated a potential difficulty with the FFT, we then show what solutions are used to make practical dynamic signal analyzers. Developing this basic knowledge of FFT characteristics makes it simple to get good results with a dynamic signal analyzer in a wide range of measurement problems.

* A more rigorous mathematical justification for the arguments developed in the main text can be found in Application Note 1405-4

Section 1: FFT Properties

The Fast Fourier Transform (FFT) is an algorithm* for *transforming* data from the time domain to the frequency domain. Since this is exactly what we want a spectrum analyzer to do, it would seem easy to implement a dynamic signal analyzer based on the FFT. However, we will see that there are many factors that complicate this seemingly straightforward task.

First, because of the many calculations involved in transforming domains, the transform must be performed on a computer if the results are to be sufficiently accurate. Fortunately, with the advent of microprocessors, it is easy and inexpensive to incorporate all the needed computing power in a small instrument package. Note, however, that we cannot now transform to the frequency domain in a continuous manner, but instead must sample and digitize the time domain input. This means that our algorithm transforms digitized samples from the time domain to samples in the frequency domain as shown in Figure 1.1.**

Because we have sampled, we no longer have an exact representation in either domain. However, a sampled representation can be as close to ideal as we desire by placing our samples closer together. Later, we will consider what sample spacing is necessary to guarantee accurate results.

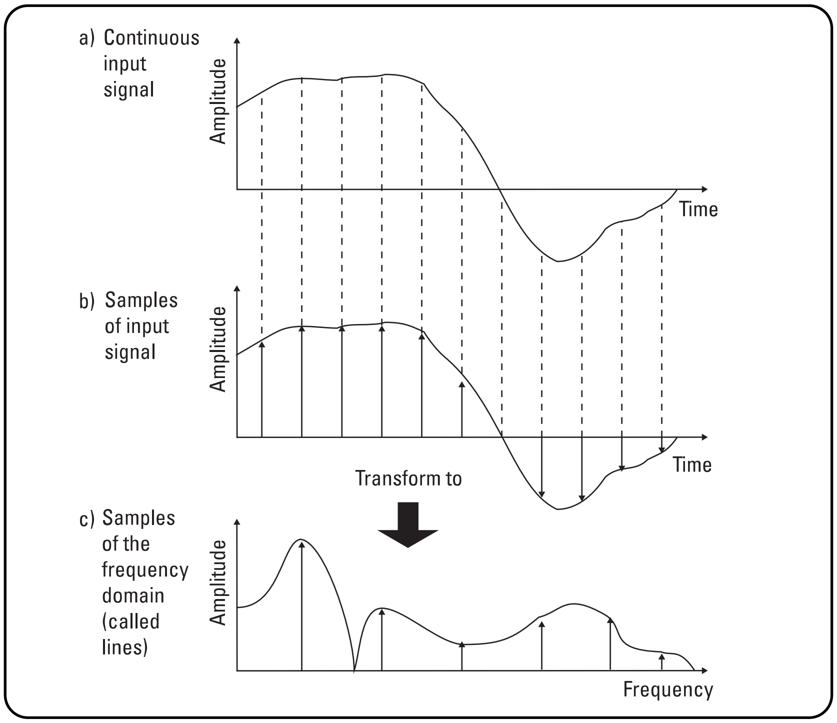


Figure 1.1. The FFT samples in both the time and frequency domains

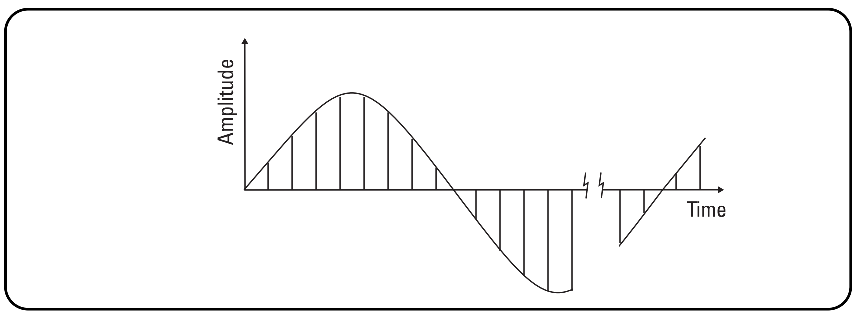


Figure 1.2. A time record is N equally spaced samples of the input.

Time Records

A *time record* is defined to be N consecutive, equally spaced samples of the input. Because it makes our transform algorithm simpler and much faster, N is restricted to be a multiple of 2, for instance 1024.

As shown in Figure 1.3, this time record is transformed as a complete *block* into a complete *block* of frequency lines. All the samples of the time record are needed to compute each and every line in the frequency domain. This is in contrast to what one might expect,

* An algorithm is any special mathematical method of solving a certain kind of problem; e.g., the technique you use to balance your checkbook.

** To reduce confusion about which domain we are in, samples in the frequency domain are called lines.

namely that a single time domain sample transforms to exactly one frequency domain line. Understanding this *block processing* property of the FFT is crucial to understanding many of the properties of the dynamic signal analyzer.

For instance, because the FFT transforms the entire time record block as a total, there cannot be valid frequency domain results until a complete time record has been gathered. However, once completed, the oldest sample could be discarded, all the samples shifted in the time record, and a new sample added to the end of the time record as in Figure 1.4. Thus, once the time record is initially filled, we have a new time record at every time domain sample and therefore could have new valid results in the frequency domain at every time domain sample.

This is very similar to the behavior of the parallel-filter analyzers described in AN1405-1, Section 3. When a signal is first applied to a parallel-filter analyzer, we must wait for the filters to respond, then we can see very rapid changes in the frequency domain. With a dynamic signal analyzer we do not get a valid result until a full time record has been gathered. Then rapid changes in the spectra can be seen.

It should be noted here that a new spectrum every sample is usually too much information, too fast. This would often give you *thousands* of transforms per second. In later sections on real-time bandwidth and overlap processing, we discuss just how fast a dynamic signal analyzer should transform.

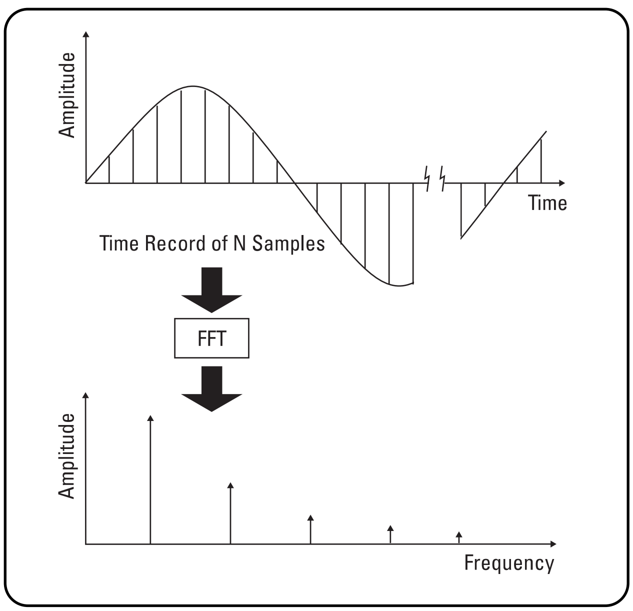


Figure 1.3. The FFT works on blocks of data.

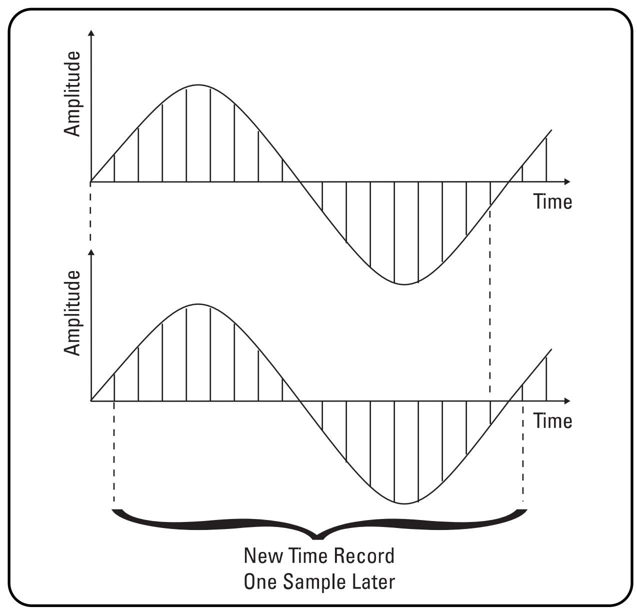


Figure 1.4. A new time record every sample after the time record is filled

How Many Lines are There?

We stated earlier that the time record has N equally spaced samples. Another property of the FFT is that it transforms these time domain samples to N/2 equally spaced lines in the frequency domain. We only get half as many lines because each frequency line actually contains two pieces of information, amplitude and phase. The meaning of this is most easily seen if we look again at the relationship between the time and frequency domain.

Figure 1.5 shows a three-dimensional graph of this relationship. Up to now, we have implied that the amplitude and frequency of the sine waves contains all the information necessary to reconstruct the input. But it should be obvious that the phase of each of these sine waves is important too. For instance, in Figure 1.6, we have shifted the *phase* of the higher frequency sine wave components of this signal. The result is a severe distortion of the original waveform.

We have not discussed the phase information contained in the spectrum of signals until now because none of the traditional spectrum analyzers are capable of measuring phase. In Application Note 1405-3, you will see that phase contains valuable information in determining the cause of performance problems.

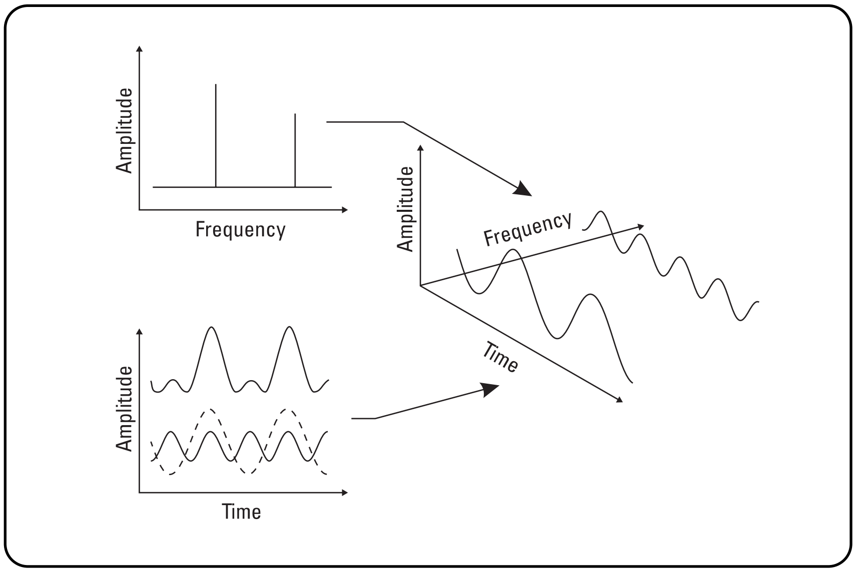


Figure 1.5. The relationship between the time and frequency domains

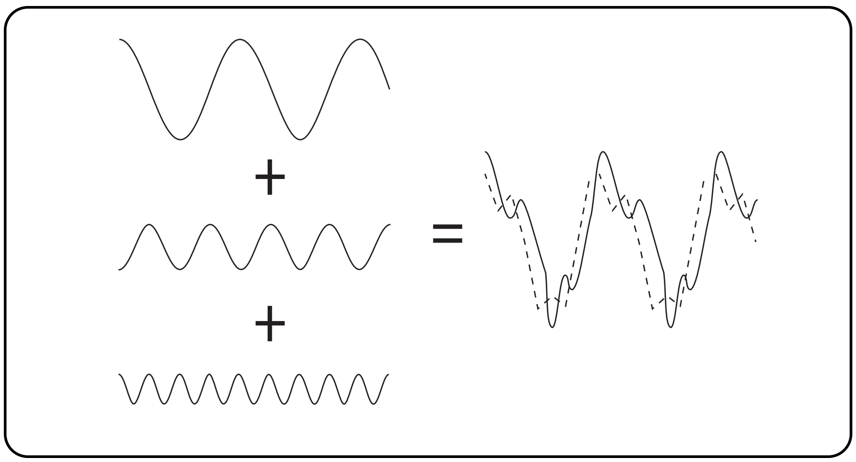


Figure 1.6. Phase of frequency domain components is important.

What is the Spacing of the Lines?

Now that we know that we have N/2 equally spaced lines in the frequency domain, what is their spacing? The lowest frequency that we can resolve with our FFT spectrum analyzer must be based on the length of the time record. We can see in Figure 1.7 that if the period of the input signal is longer than the time record, we have no way of determining the period (or frequency, its reciprocal). Therefore, the lowest frequency line of the FFT must occur at frequency equal to the reciprocal of the time record length.

In addition, there is a frequency line at zero Hertz, dc. This is merely the average of the input over the time record. It is rarely used in spectrum or network analysis. But, we have now established the spacing between these two lines and hence every line; it is the reciprocal of the time record.

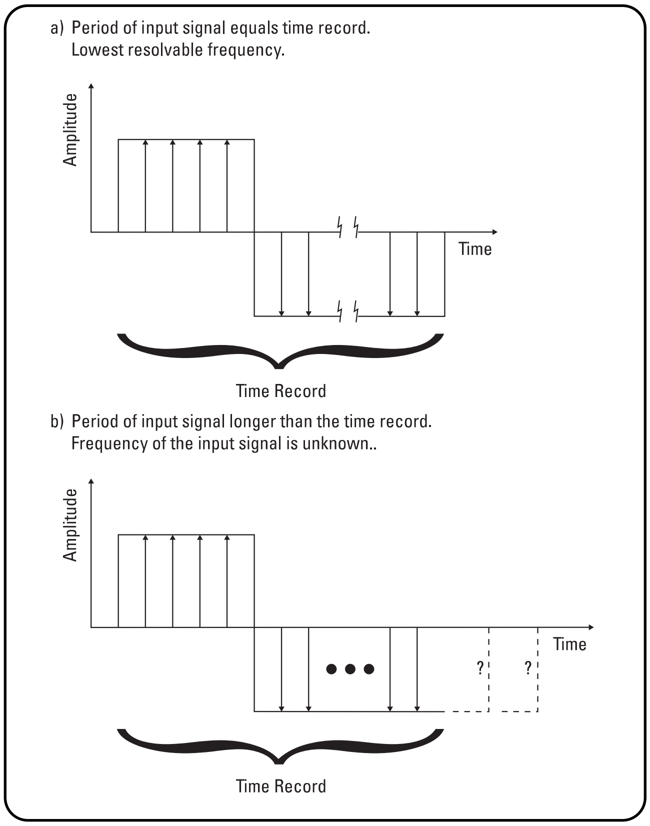


Figure 1.7. Lowest frequency resolvable by the FFT

What is the Frequency Range of the FFT?

We can now quickly determine that the highest frequency we can measure is:

$$f_{max} = \frac{N}{2}\left(\frac{1}{\text{Period of Time Record}}\right)$$

because we have N/2 lines spaced by the reciprocal of the time record starting at zero Hertz.*

Since we would like to adjust the frequency range of our measurement, we must vary fmax. The number of time samples N is fixed by the implementation of the FFT algorithm. Therefore, we must vary the period of the time record to vary fmax. To do this, we must vary the sample rate so that we always have N samples in our variable time record period. This is illustrated in Figure 1.9. Notice that to cover higher frequencies, we must sample faster.

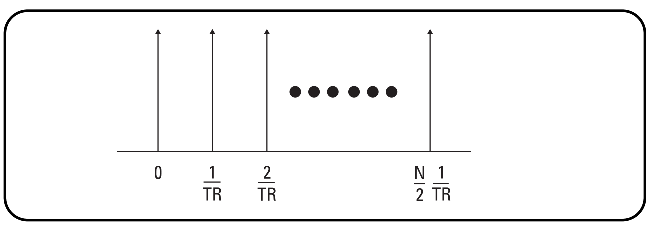


Figure 1.8. Frequencies of all the spectral lines of the FFT

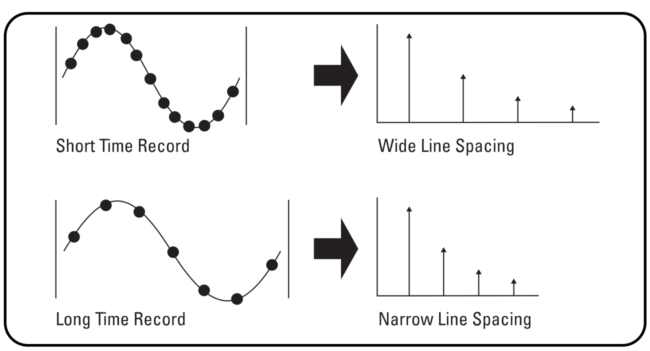


Figure 1.9. Frequency range of dynamic signal analyzers is determined by sample rate.

* The usefulness of this frequency range can be limited by the problem of aliasing. Aliasing is discussed in Section 3.

Section 2:* Sampling and Digitizing

Recall that the input to our dynamic signal analyzer is a continuous analog voltage. This voltage might be from an electronic circuit or it could be the output of a transducer and be proportional to current, power, pressure, acceleration or any number of other inputs. Recall also that the FFT requires digitized samples of the input for its digital calculations. Therefore, we need to add a sampler and analog-to-digital converter (ADC) to our FFT processor to make a spectrum analyzer. We show this basic block diagram in Figure 2.1.

For the analyzer to have the high accuracy needed for many measurements, the sampler and ADC must be quite good. The sampler must sample the input at exactly the correct time and must accurately hold the input voltage measured at this time until the ADC has finished its conversion. The ADC must have high resolution and linearity. For 70 dB of dynamic range the ADC must have at least 12 bits of resolution and one half least-significant-bit linearity.

A good digital multimeter (DMM) will typically exceed these specifications, but the ADC for a dynamic signal analyzer must be much faster than typical fast DMMs. A fast DMM might take a thousand readings per second, but in a typical dynamic signal analyzer the ADC must take at least a hundred thousand readings per second.

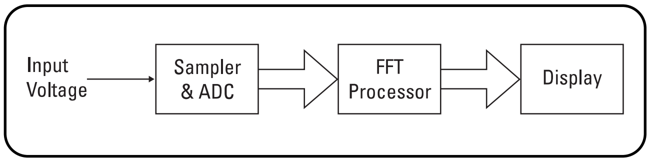


Figure 2.1. Block diagram of a dynamic signal analyzer

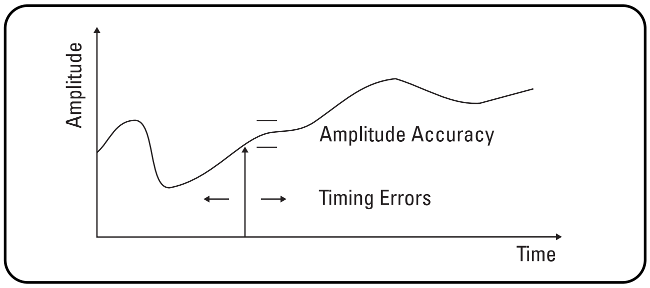


Figure 2.2. The sampler and ADC must not introduce errors.

* You can skip this section and the next if you are not interested in the internal operation of a dynamic signal analyzer. However, if you specify the purchase of dynamic signal analyzers, you are especially encouraged to read these sections. The basic knowledge you gain from these sections can insure you specify the best analyzer for your requirements.

Section 3: Aliasing

The reason an FFT spectrum analyzer needs so many samples per second is to avoid a problem called aliasing. Aliasing is a potential problem in any sampled data system. It is often overlooked, sometimes with disastrous results.

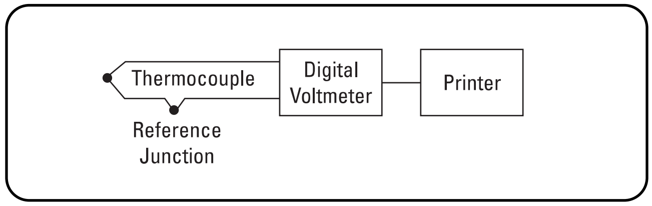


Figure 3.1. A simple sampled data system

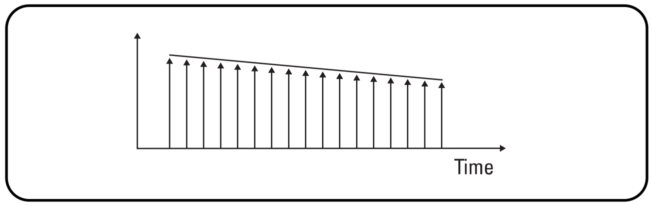


Figure 3.2. Plot of temperature variation of a room

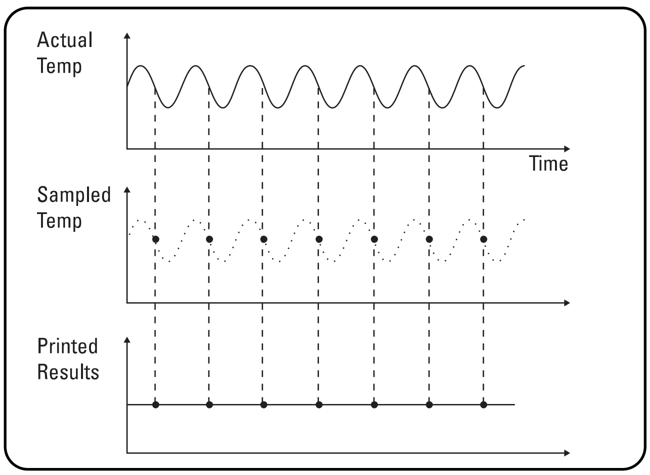


Figure 3.3. Plot of temperature variation of a small part

A Simple Data Logging Example of Aliasing

Let us look at a simple data logging example to see what aliasing is and how it can be avoided. Consider the example for recording temperature shown in Figure 3.1. A thermocouple is connected to a digital voltmeter that is in turn connected to a printer. The system is set up to print the temperature every second. What would we expect for an output? If we were measuring the temperature of a room that only changes slowly, we would expect every reading to be almost the same as the previous one. In fact, we are sampling much more often than necessary to determine the temperature of the room with time. If we plotted the results of this "thought experiment," we would expect to see results like Figure 3.2.

The Case of the Missing Temperature

If, on the other hand, we were measuring the temperature of a small part that could heat and cool rapidly, what would the output be? Suppose that the temperature of our part cycled exactly once every second. As shown in Figure 3.3, our printout says that the temperature never changes.

What has happened is that we have sampled at exactly the same point on our periodic temperature cycle with every sample. We have not sampled fast enough to see the temperature fluctuations.

Aliasing in the Frequency Domain

This completely erroneous result is due to a phenomena called aliasing.* Aliasing is shown in the frequency domain in Figure 3.4. Two signals are said to alias if the difference of their frequencies falls in the frequency range of interest. This difference frequency is always generated in the process of sampling. In Figure 3.4, the input frequency is slightly higher than the sampling frequency so a low frequency alias term is generated. If the input frequency equals the sampling frequency as in our small part example, then the alias term falls at DC (zero Hertz) and we get the constant output that we saw above.

Aliasing is not always bad. It is called mixing or heterodyning in analog electronics, and is commonly used for tuning household radios and televisions as well as many other communication products. However, in the case of the missing temperature variation of our small part, we definitely have a problem. How can we guarantee that we will avoid this problem in a measurement situation?

Figure 3.5 shows that if we sample at greater than twice the highest frequency of our input, the alias products will not fall within the frequency range of our input. Therefore, a filter (or our FFT processor, which acts like a filter) after the sampler will remove the alias products while passing the desired input signals *if the sample rate is greater than twice the highest frequency of the input*. If the sample rate is lower, the alias products will fall in the frequency range of the input and no amount of filtering will be able to remove them from the signal.

* Aliasing is also known as fold-over or mixing.

This minimum sample rate requirement is known as the Nyquist Criterion. It is easy to see in the time domain that a sampling frequency exactly twice the input frequency would not always be enough. It is less obvious that slightly more than two samples in each period is sufficient information. It certainly would not be enough to give a high-quality time display. Yet we saw in Figure 3.5 that meeting the Nyquist Criterion of a sample rate greater than twice the maximum input frequency is sufficient to avoid aliasing and preserve all the information in the input signal.

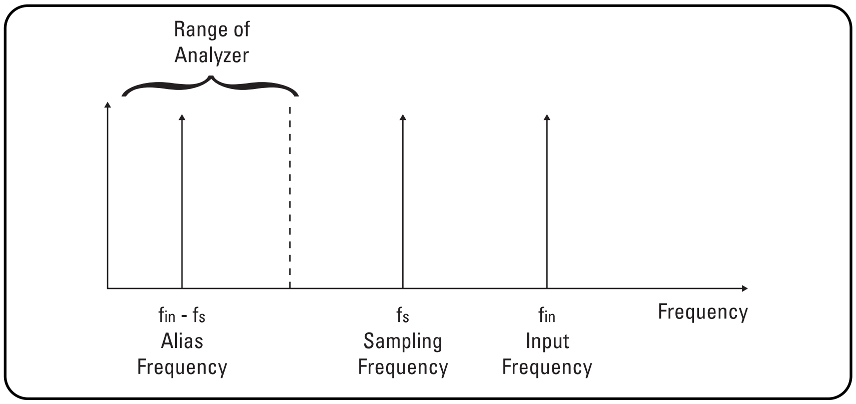


Figure 3.4. The problem of aliasing viewed in the frequency domain

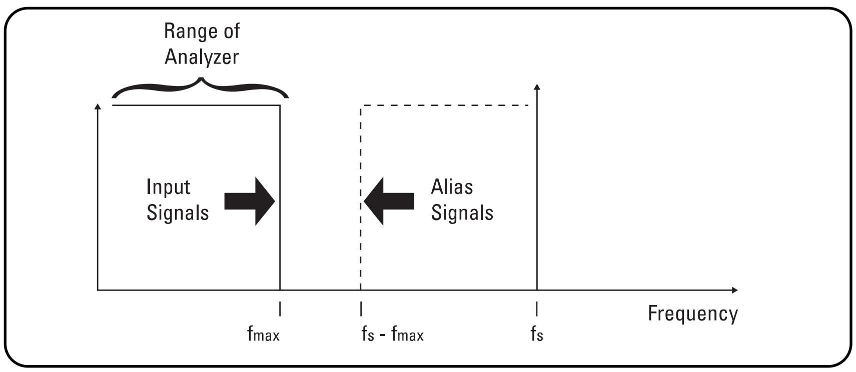


Figure 3.5. A frequency domain view of how to avoid aliasing - sample at greater than twice the highest input frequency

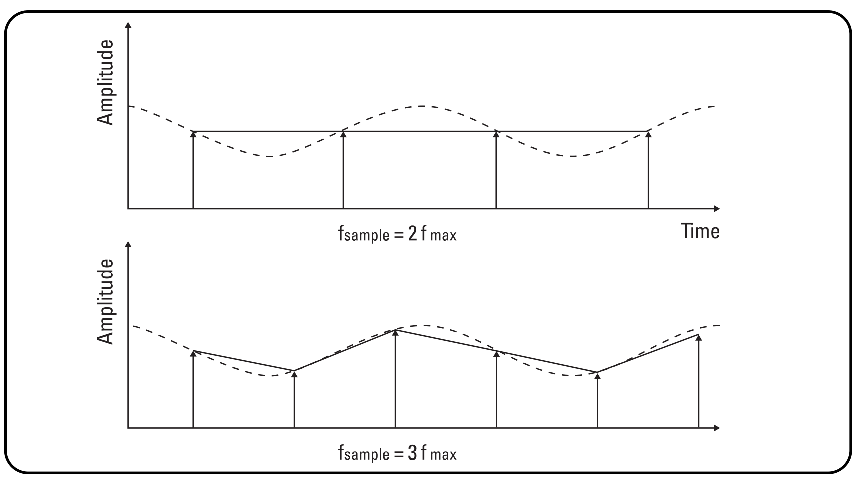


Figure 3.6. Nyquist Criterion in the time domain

The Need for an Anti-Alias Filter

Unfortunately, the real world rarely restricts the frequency range of its signals. In the case of the room temperature, we can be reasonably sure of the maximum rate at which the temperature could change, but we still can not rule out stray signals. Signals induced at the power-line frequency or even local radio stations could alias into the desired frequency range. The only way to be really certain that the input frequency range is limited is to add a low pass filter before the sampler and ADC. Such a filter is called an anti-alias filter.

An ideal anti-alias filter would look like Figure 3.7a. It would pass all the desired input frequencies with no loss and completely reject any higher frequencies which otherwise could alias into the input frequency range. However, it is not even theoretically possible to build such a filter, much less practical. Instead, all real filters look something like Figure 3.7b with a gradual roll off and finite rejection of undesired signals. Large input signals that are not well attenuated in the transition band could still alias into the desired input frequency range. To avoid this, the sampling frequency is raised to twice the highest frequency of the transition band. This guarantees that any signals that could alias are well attentuated by the stop band of the filter. Typically, this means that the sample rate is now two-and-a-half to four times the maximum desired input frequency. Therefore, a 25 kHz FFT spectrum analyzer can require an ADC that runs at 100 kHz*.

* Unfortunately, because the spacing of the FFT lines depends on the sample rate, increasing the sample rate decreases the number of lines that are in the desired frequency range. Therefore, to avoid aliasing problems dynamic signal analyzers have only .25N to .4N lines instead of N/2 lines.

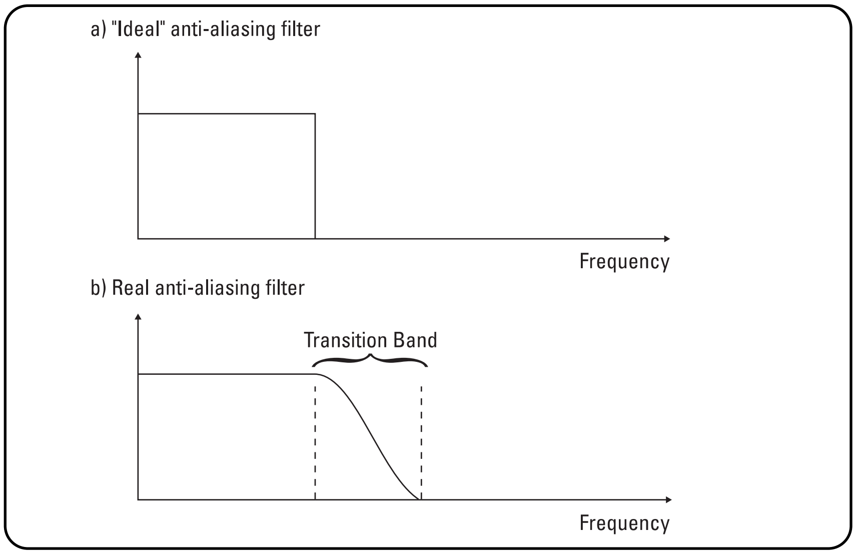


Figure 3.7. Actual anti-alias filters require higher sampling frequencies.

The Need for More Than One Anti-Alias Filter

You may recall from Section 1 that due to the properties of the FFT, we must vary the sample rate to vary the frequency span of our analyzer. To reduce the frequency span, we must reduce the sample rate. From our considerations of aliasing, we now realize that we must also reduce the anti-alias filter frequency by the same amount.

Since a dynamic signal analyzer is a very versatile instrument used in a wide range of applications, it is desirable to have a wide range of frequency spans available. Typical instruments have a minimum span of 1 Hertz and a maximum of tens to hundreds of kilohertz. This four-decade range typically needs to be covered with at least three spans per decade. This would mean at least twelve anti-alias filters would be required for each channel.

Each of these filters must have very good performance. Their transition bands should be as narrow as possible, so that as many lines as possible are free from alias products. Additionally, in a two-channel analyzer, each filter pair must be well matched for accurate network analysis measurements. These two points, unfortunately, mean that each of the filters is expensive. Taken together they can add significantly to the price of the analyzer. To cut expenses, some manufacturers don't use a low enough frequency anti-alias filter on the lowest-frequency spans. (The lowest frequency filters cost the most of all.) But as we have seen, this can lead to problems like our "case of the missing temperature."

Digital Filtering

Fortunately, there is an alternative that is cheaper, and when used in conjunction with a single analog anti-alias filter, always provides aliasing protection. It is called “digital filtering,” because it filters the input signal after we have sampled and digitized it. To see how this works, let us look at Figure 3.8.

In the analog case we already discussed, we had to use a new filter every time we changed the sample rate of the analog-to-digital converter (ADC). For digital filtering, the ADC sample rate is left constant at the rate needed for the highest-frequency span of the analyzer. This means we need not change our anti-alias filter. To get the reduced sample rate and filtering we need for the narrower frequency spans, we follow the ADC with a digital filter.

This digital filter is known as a decimating filter. It not only filters the digital representation of the signal to the desired frequency span, it also reduces the sample rate at its output to the rate needed for that frequency span. Because this filter is digital, there are no manufacturing variations, aging or drift in the filter. Therefore, in a two-channel analyzer, the filters in each channel are identical. It is easy to design a single digital filter to work on many frequency spans so the need for multiple filters per channel is avoided. All these factors taken together mean that digital filtering is much less expensive than analog anti-aliasing filtering.

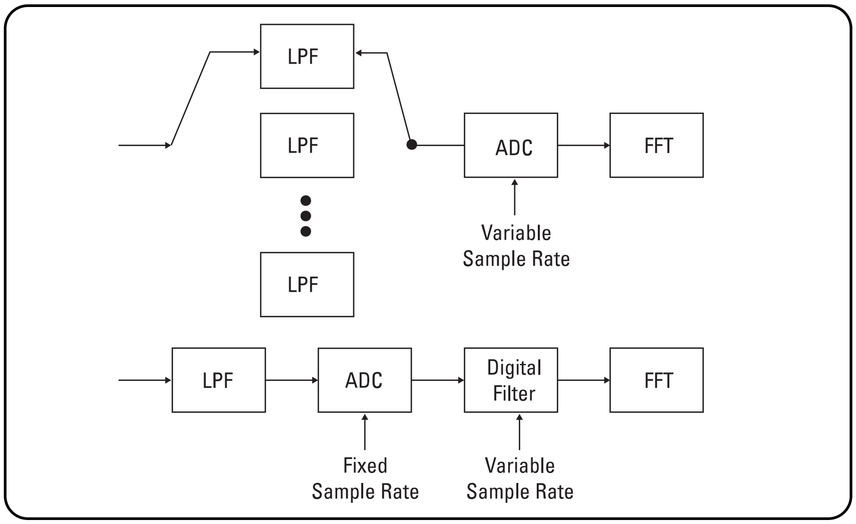


Figure 3.8. Block diagrams of analog and digital filtering

Section 4: Band-Selectable Analysis

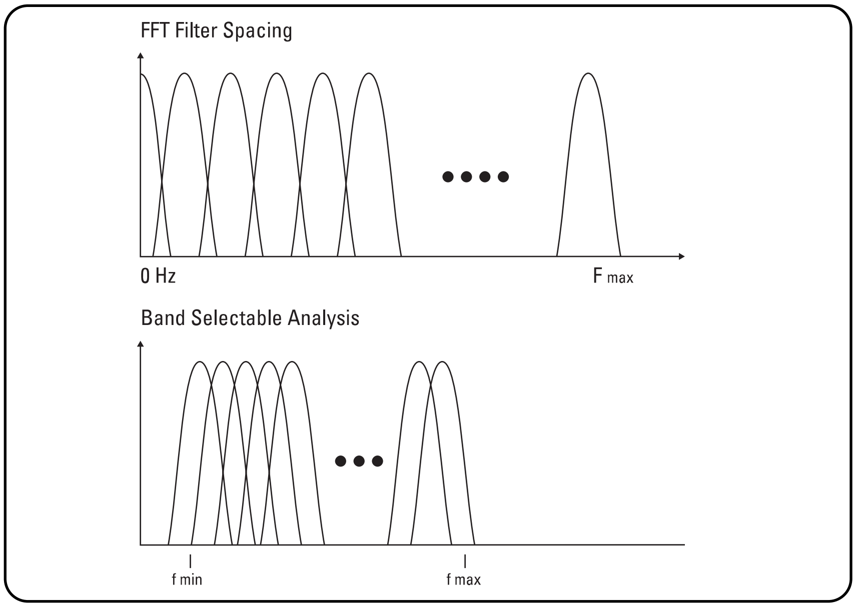


Figure 4.1. High-resolution measurements with band-selectable analysis

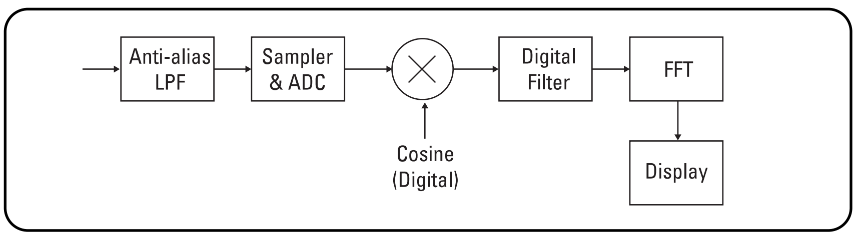


Figure 4.2. Analyzer block diagram

Suppose we need to measure a small signal that is very close in frequency to a large one. We might be measuring the power-line side-bands (50 or 60 Hz) on a 20 kHz oscillator. Or we might want to distinguish between the stator vibration and the shaft imbalance in the spectrum of a motor.*

Recall from our discussion of the properties of the Fast Fourier Transform that it is equivalent to a set of filters, starting at zero Hertz, equally spaced up to some maximum frequency. Therefore, our frequency resolution is limited to the maximum frequency divided by the number of filters.

To just resolve the 60 Hz sidebands on a 20 kHz oscillator signal would require 333 lines (or filters) of the FFT. Two or three times more lines would be required to accurately measure the sidebands. But typical dynamic signal analyzers only have 200 to 400 lines, not enough for accurate measurements. To increase the number of lines would greatly increase the cost of the analyzer. If we chose to pay the extra cost, we would still have trouble seeing the results. With a 4-inch (10 cm) screen, the sidebands would be only 0.01 inch (.25 mm) from the carrier.

A better way to solve this problem is to concentrate the filters into the frequency range of interest as in Figure 4.1. If we select the minimum frequency as well as the maximum frequency of our filters we can "zoom in" for a high resolution close-up shot of our frequency spectrum. We now have the capability of looking at the entire spectrum at once with low resolution, as well as the ability to look at what interests us with much higher resolution.

This capability of increased resolution is called band-selectable analysis (BSA).** It is done by mixing or heterodyning the input signal down into the range of the FFT span selected. This technique, familiar to electronic engineers, is the process by which radios and televisions tune in stations.

The primary difference between the implementation of BSA in dynamic signal analyzers and heterodyne radios is shown in Figure 4.2. In a radio, the sine wave used for mixing is an analog voltage. In a dynamic signal analyzer, the mixing is done after the input has been digitized, so the "sine wave" is a series of digital numbers into a digital multiplier. This means that the mixing will be done with a very accurate and stable digital signal so our high-resolution display will likewise be very stable and accurate.

* The shaft of an ac induction motor always runs at a rate slightly lower than a multiple of the driven frequency, an effect called slippage.

** Also sometimes called "zoom."

Section 5: Windowing

The Need for Windowing

There is another property of the Fast Fourier Transform that affects its use in frequency domain analysis. We recall that the FFT computes the frequency spectrum from a block of samples of the input called a time record. In addition, the FFT algorithm is based upon the assumption that this time record is repeated throughout time, as illustrated in Figure 5.1.

This does not cause a problem with the transient case shown. But what happens if we are measuring a continuous signal like a sine wave? If the time record contains an integral number of cycles of the input sine wave, then this assumption exactly matches the actual input waveform as shown in Figure 5.2. In this case, the input waveform is said to be *periodic* in the time record.

Figure 5.3 demonstrates the difficulty with this assumption when the input is not periodic in the time record. The FFT algorithm is computed on the basis of the highly distorted waveform in Figure 5.3c. We know from Chapter 2 that the actual sine wave input has a frequency spectrum of single line. The spectrum of the input assumed by the FFT in Figure 5.3c should be very different. Since sharp phenomena in one domain are spread out in the other domain, we would expect the spectrum of our sine wave to be spread out through the frequency domain.

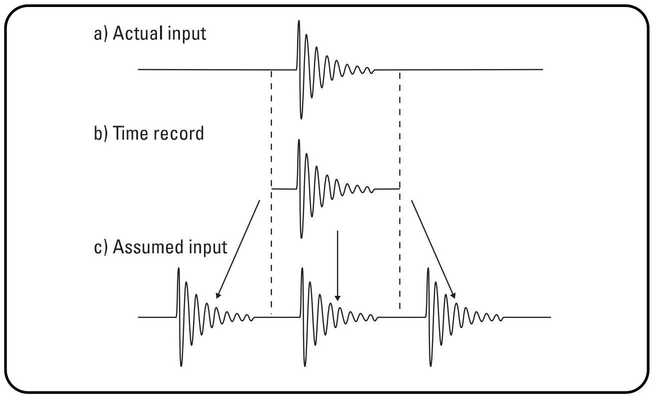


Figure 5.1. FFT assumption — time record repeated throughout all time

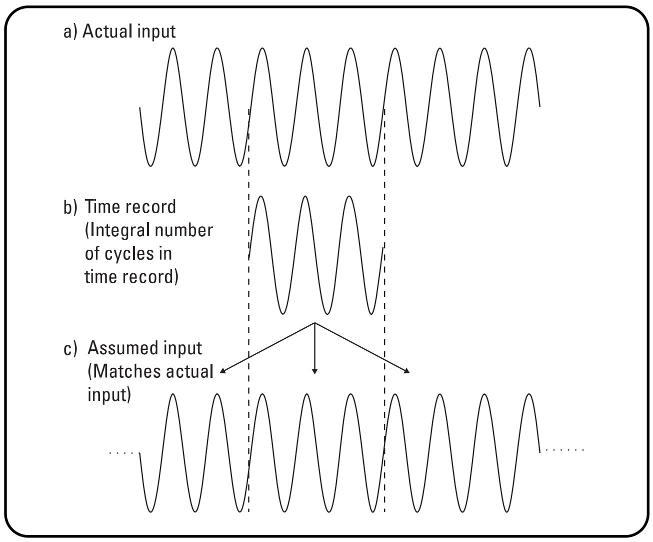


Figure 5.2. Input signal periodic in time record

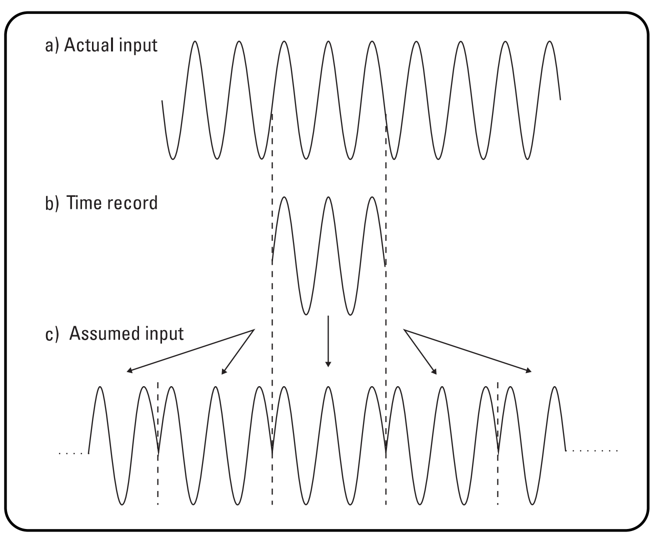


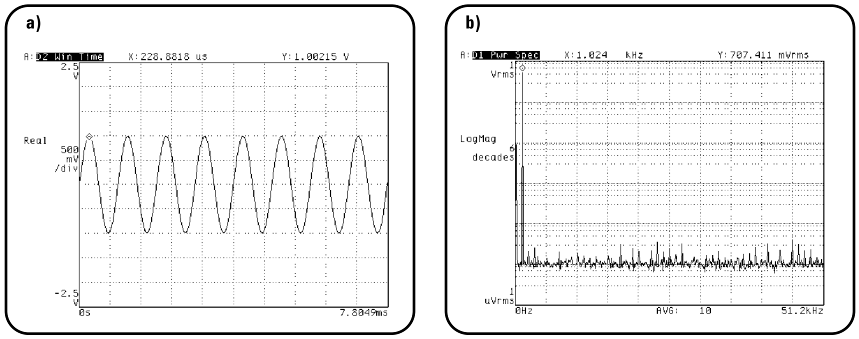
Figure 5.3. Input signal not periodic in time record

In Figure 5.4 we see in an actual measurement that our expectations are correct. In Figures 5.4a and b, we see a sine wave that is periodic in the time record. Its frequency spectrum is a single line whose width is determined only by the resolution of our dynamic signal analyzer. On the other hand, Figures 5.4c and d show a sine wave that is not periodic in the time record. Its power has been spread throughout the spectrum as we predicted.

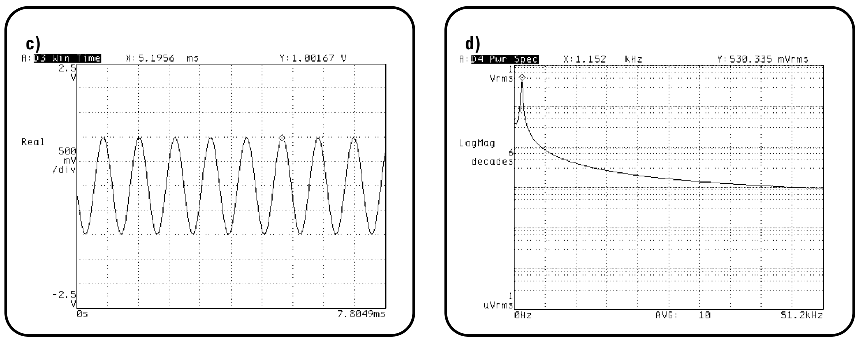
This smearing of energy throughout the frequency domains is a phenomena known as *leakage*. We are seeing energy leak out of one resolution line of the FFT into all the other lines.

It is important to realize that leakage is due to the fact that we have taken a finite time record. For a sine wave to have a single line spectrum, it must exist for all time, from minus infinity to plus infinity. If we were to have an infinite time record, the FFT would compute the correct single line spectrum exactly. However, since we are not willing to wait forever to measure its spectrum, we only look at a finite time record of the sine wave. This can cause leakage if the continuous input is not periodic in the time record.

It is obvious from Figure 5.4 that the problem of leakage is severe enough to entirely mask small signals close to our sine waves. As such, the FFT would not be a very useful spectrum analyzer. The solution to this problem is known as windowing. The problems of leakage and how to solve them with windowing can be the most confusing concepts of dynamic signal analysis. Therefore, we will now carefully develop the problem and its solution in several representative cases.



a) and b) Sine wave periodic in time record



c) and d) Sine wave not periodic in time record

Figure 5.4. Actual FFT results

What is Windowing?

In Figure 5.5 we have again reproduced the assumed input wave form of a sine wave that is not periodic in the time record. Notice that most of the problem seems to be at the edges of the time record; the center is a good sine wave. If the FFT could be made to ignore the ends and concentrate on the middle of the time record, we would expect to get much closer to the correct single-line spectrum in the frequency domain.

If we multiply our time record by a function that is zero at the ends of the time record and large in the middle, we would concentrate the FFT on the middle of the time record. One such function is shown in Figure 5.5c. Such functions are called window functions because they force us to look at data through a narrow window.

Figure 5.6 shows us the vast improvement we get by windowing data that is not periodic in the time record. However, it is important to realize that we have tampered with the input data and cannot expect perfect results. The FFT assumes the input looks like Figure 5.5d, something like an amplitude-modulated sine wave. This has a frequency spectrum which is closer to the correct single line of the input sine wave than Figure 5.5b, but it still is not correct. Figure 5.7 demonstrates that the windowed data does not have as narrow a spectrum as an unwindowed function which is periodic in the time record.

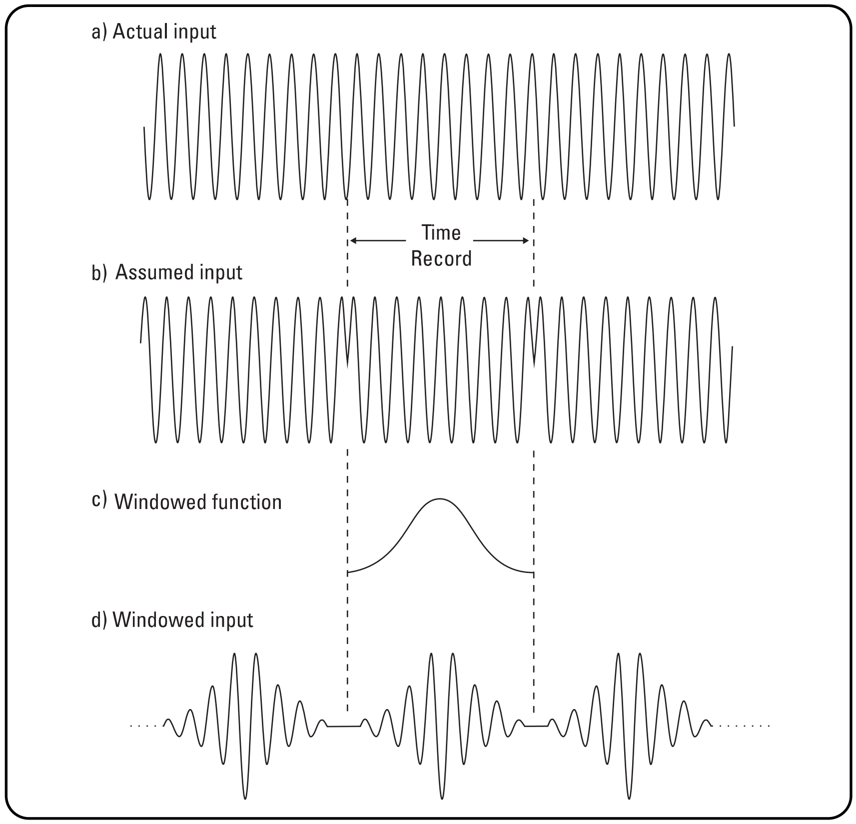
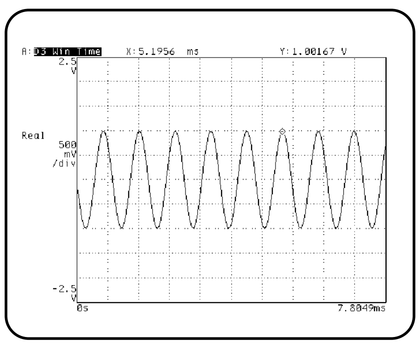
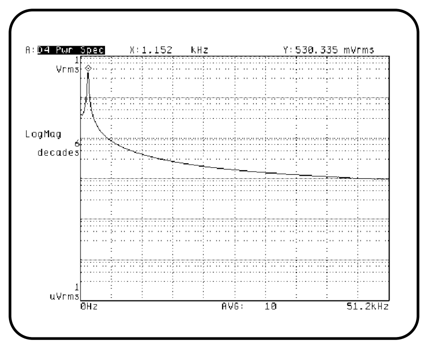


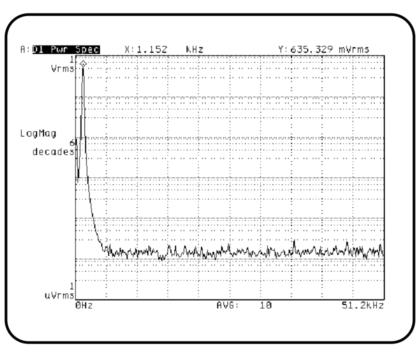
Figure 5.5. The effect of windowing in the time domain



a) Sine wave not periodic in time record



b) FFT results with no window function

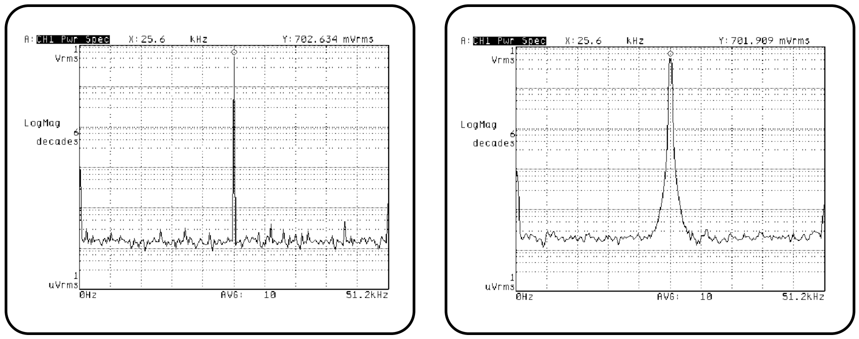


c) FFT results with a window function

Figure 5.6. Leakage reduction with windowing

The Hanning Window

Any number of functions can be used to window the data, but the most common one is called Hanning. We actually used the Hanning window in Figure 5.6 as our example of leakage reduction with windowing. The Hanning window is also commonly used when measuring random noise.



a) Leakage-free measurement — input periodic in time record

b) Windowed measurement — input not periodic in time record

Figure 5.7. Windowing reduces leakage but does not eliminate it.

The Uniform Window*

We have seen that the Hanning window does an acceptably good job on our sine wave examples, both periodic and non-periodic in the time record. If this is true, why should we want any other windows?

Suppose that instead of wanting the frequency spectrum of a continuous signal, we would like the spectrum of a transient event. A typical transient is shown in Figure 5.8a. If we multiplied it by the window function in Figure 5.8b we would get the highly distorted signal shown in Figure 5.8c. The frequency spectrum of an actual transient with and without the Hanning window is shown in Figure 5.9. The Hanning window has taken our transient, which naturally has energy spread widely through the frequency domain and made it look more like a sine wave.

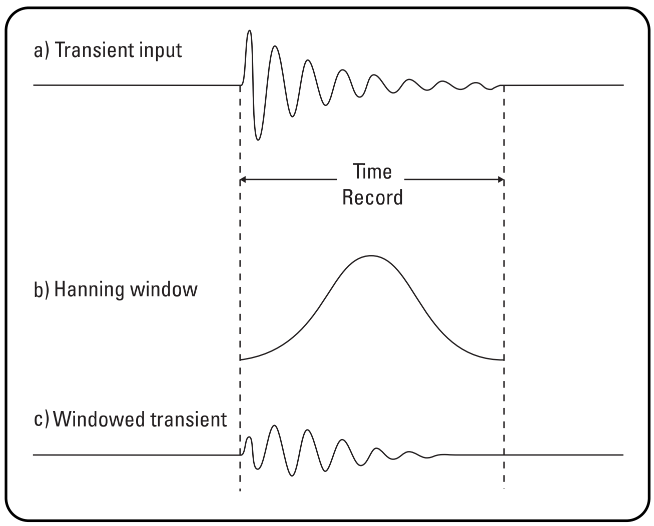
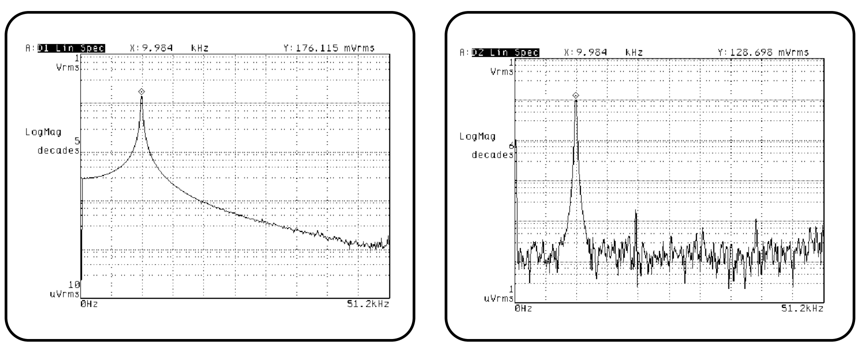


Figure 5.8. Windowing loses information from transient events.



a) Unwindowed trainsients

b) Hanning windowed transients

Figure 5.9. Spectrums of transients

Therefore, we can see that for transients we do not want to use the Hanning window. We would like to use all the data in the time record equally or uniformly. Hence we will use a *uniform window* which weights all of the time record uniformly.

The case we made for the uniform window by looking at transients can be generalized. Notice that our transient has the property that it is zero at the beginning and end of the time record. Remember that we introduced windowing to force the input to be zero at the *ends of the time record*. In this case, there is no need for windowing the input. Any function like this which does not require a window because it occurs completely within the time record is called a *self-windowing function*. Self-windowing functions generate no leakage in the FFT and so need no window.

* The uniform window is sometimes referred to as a "rectangular window."

There are many examples of self-windowing functions, some of which are shown in Figure 5.10. Impacts, impulses, shock responses, sine bursts, noise bursts, chirp bursts and pseudo-random noise can all be made to be self-windowing. Self-windowing functions are often used as the excitation in measuring the frequency response of networks, particularly if the network has lightly-damped resonances (high Q). This is because the self-windowing functions generate no leakage in the FFT. Recall that even with the Hanning window, some leakage was present when the signal was not periodic in the time record. This means that without a self-windowing excitation, energy could leak from a lightly damped resonance into adjacent lines (filters). The resulting spectrum would show greater damping than actually exists.*

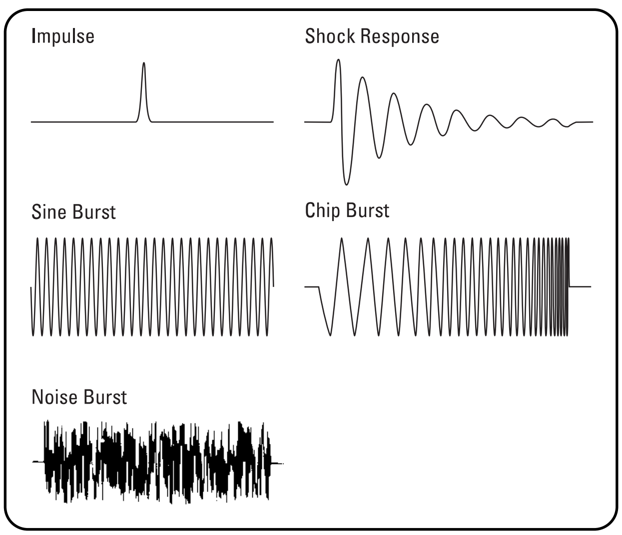


Figure 5.10. Self-windowing function examples

The Flat-top Window

We have shown that we need a uniform window for analyzing self-windowing functions like transients. In addition, we need a Hanning window for measuring noise and periodic signals like sine waves.

We now need to introduce a third window function, the *flat-top window*, to avoid a subtle effect of the Hanning window. To understand this effect, we need to look at the Hanning window in the frequency domain. We recall that the FFT acts like a set of parallel filters. Figure 5.11 shows the shape of those filters when the Hanning window is used. Notice that the Hanning function gives the filter a very rounded top. If a component of the input signal is centered in the filter it will be measured accurately.** Otherwise, the filter shape will attenuate the component by up to 1.5 dB (16 percent) when it falls midway between the filters.

This error is unacceptably large if we are trying to measure a signal's amplitude accurately. The solution is to choose a window function which gives the filter a flatter passband. Such a flat-top passband shape is shown in Figure 5.12. The amplitude error from this window function does not exceed .1 dB (1%), a 1.4 dB improvement.

The accuracy improvement does not come without its price, however. Figure 5.13 shows that we have flattened the top of the

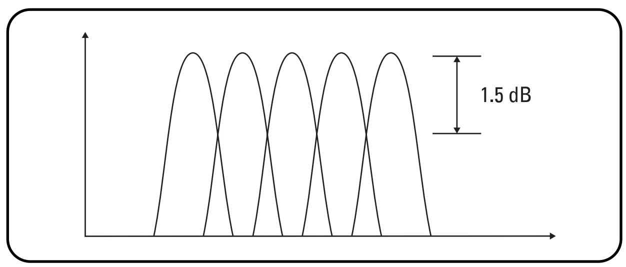


Figure 5.11. Hanning passband shapes

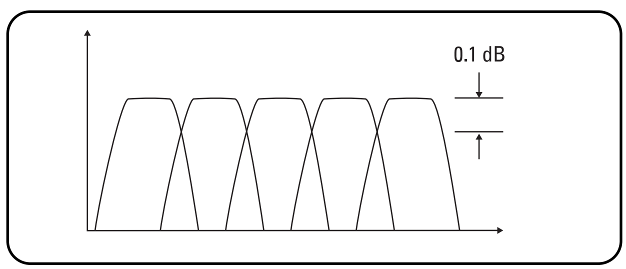


Figure 5.12. Flat-top passband shapes

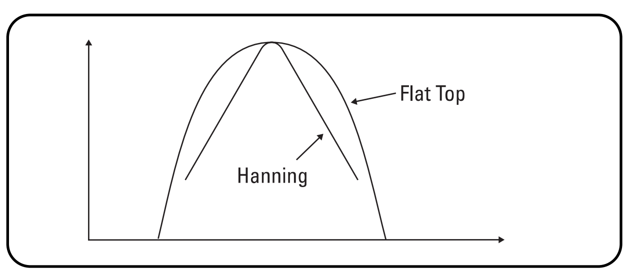


Figure 5.13. Reduced resolution of the flat-top window

* There is another way to avoid this problem using band-selectable analysis. We illustrate this in Agilent Application Note 1405-3.

** It will, in fact, be periodic in the time record.

passband at the expense of widening the skirts of the filter. We therefore lose some ability to resolve a small component, closely spaced to a large one. Some dynamic signal analyzers offer both Hanning and flat-top window functions so that the operator can choose between increased accuracy or improved frequency resolution.

Other Window Functions

Many other window functions are possible but the three listed above are by far the most common for general measurements. For special measurement situations other groups of window functions may be useful. We will discuss two windows that are particularly useful when doing network analysis on mechanical structures by impact testing.

The Force and Response Windows

A hammer equipped with a force transducer is commonly used to stimulate a structure for response measurements. Typically the force input is connected to one channel of the analyzer and the response of the structure from another transducer is connected to the second channel. This force impact is obviously a self-windowing function. The response of the structure is also self-windowing if it dies out within the time record of the analyzer. To guarantee that the response does go to zero by the end of the time record, an exponential-weighted window called a response window is sometimes added. Figure 5.14 shows a response window acting on the response of a lightly damped structure which did not fully decay by the end of the time record. Notice that unlike the Hanning window, the response window is not zero at both ends of the time record. We know that the response of the structure will be zero at the beginning of the time record (before the hammer blow) so there is no need for the window function to be zero there. In addition, most of the information about the structural response is contained at the beginning of the time record so we make sure that this is weighted most heavily by our response window function.

The time record of the exciting force should be just the impact with the structure. However, movement of the hammer before and after hitting the structure can cause stray signals in the time record. One way to avoid this is to use a force window shown in Figure 5.15. The force window is unity where the impact data is valid and zero everywhere else so that the analyzer does not measure any stray noise that might be present.

Passband Shapes or Window Functions?

In the proceeding discussion we sometimes talked about window functions in the time domain. At other times we talked about the filter passband shape in the frequency domain caused by these windows. We change our perspective freely to whichever domain yields the simplest explanation. Likewise, some dynamic signal analyzers call the uniform, Hanning and flat-top functions "windows" and other analyzers call those functions "pass-band shapes." Use whichever terminology is easier for the problem at hand, as they are completely interchangeable, just as the time and frequency domains are completely equivalent.

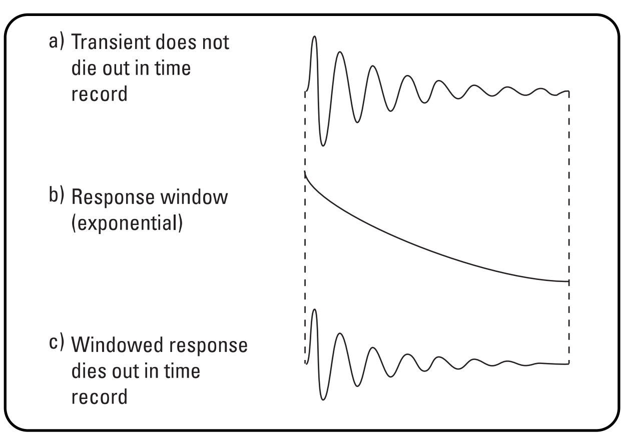


Figure 5.14. Using the response window

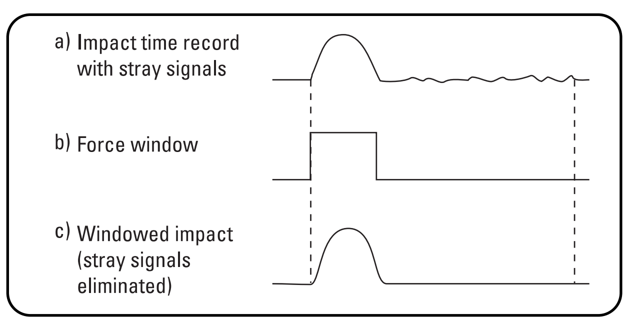


Figure 5.15. Using the force window

Section 6: Network Stimulus

As we explained in Agilent Application Note 1405-1, we can measure the frequency response at one frequency by stimulating the network with a single sine wave and measuring the gain and phase shift at that frequency. The frequency of the stimulus is then changed and the measurement repeated until all desired frequencies have been measured. Every time the frequency is changed, the network response must settle to its steady-state value before a new measurement can be taken, making this measurement process a slow task.

Many network analyzers operate in this manner and we can make the measurement this way with a two-channel dynamic signal analyzer. We set the sine wave source to the center of the first filter as in Figure 6.1. The analyzer then measures the gain and phase of the network at this frequency while the rest of the analyzer's filters measure only noise. We then increase the source frequency to the next filter center, wait for the network to settle and then measure the gain and phase. We continue this procedure until we have measured the gain and phase of the network at all the frequencies of the filters in our analyzer.

This procedure would, within experimental error, give us the same results as we would get with any of the network analyzers described in Agilent Application Note 1405-1, *with any network, linear or nonlinear*.

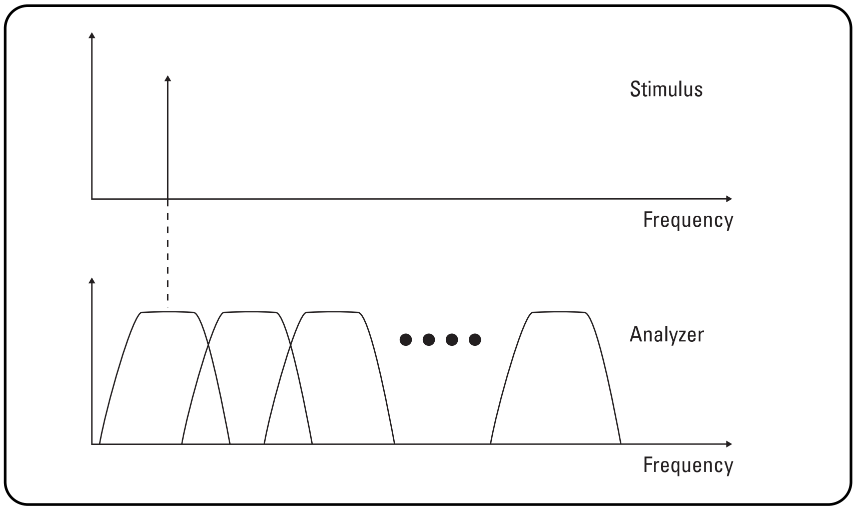


Figure 6.1. Frequency response measurements with a sine wave stimulus

Noise as a Stimulus

A single sine wave stimulus does not take advantage of the possible speed the parallel filters of a dynamic signal analyzer provide. If we had a source that put out multiple sine waves, each one centered in a filter, then we could measure the frequency response at all frequencies at one time. Such a source, shown in Figure 6.2, acts like hundreds of sine wave generators connected together. Although this sounds very expensive, just such a source can be easily generated digitally. It is called a pseudo-random noise or periodic random noise source.

From the names used for this source it is apparent that it acts somewhat like a true noise generator, except that it has periodicity. If we add together a large number of sine waves, the result is very much like white noise. A good analogy is the sound of rain. A single drop of water makes a quite distinctive splashing sound, but a rain storm sounds like white noise. However, if we add together a large number of sine waves, our noise-like signal will periodically repeat its sequence. Hence, the name *periodic random noise* (PRN) source.

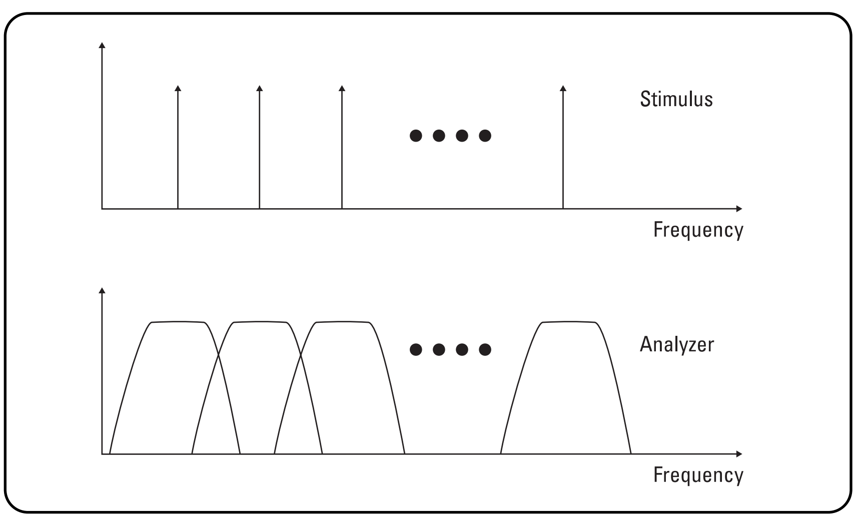


Figure 6.2. Pseudo-random noise as a stimulus

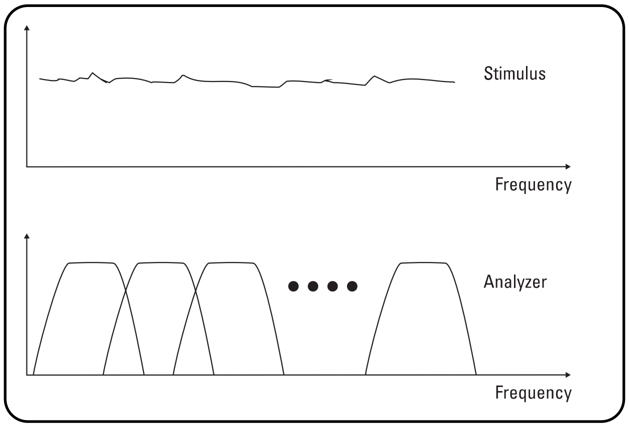


Figure 6.3. Random noise as a stimulus

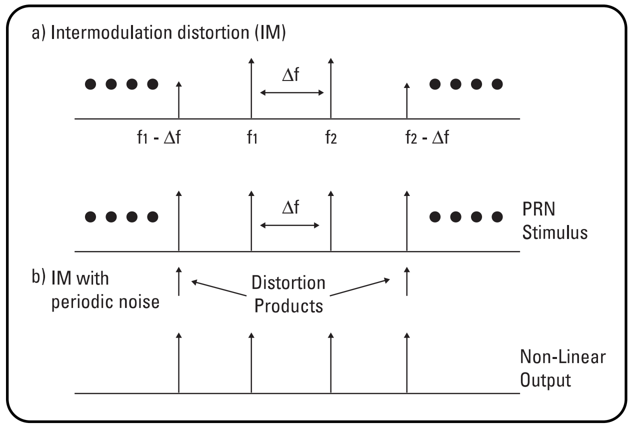
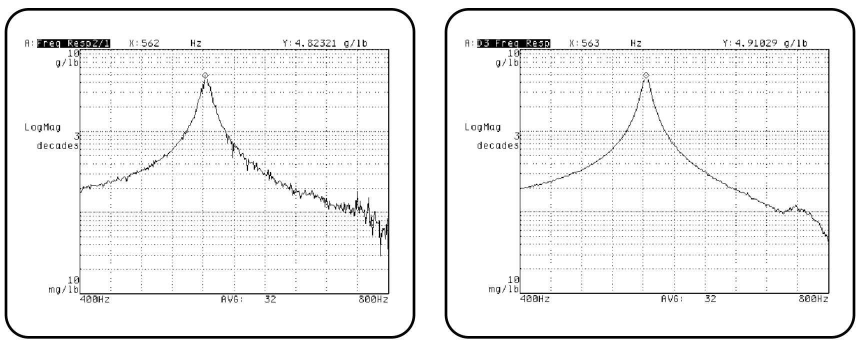


Figure 6.4. Pseudo-random noise distortion

A truly random noise source has a spectrum shown in Figure 6.3. It is apparent that a random noise source would also stimulate all the filters at one time and so could be used as a network stimulus. Which is a better stimulus? The answer depends upon the measurement situation.



a) Pseudo-random noise stimulus

b) Random noise stimulus

Figure 6.5. Nonlinear transfer function

Linear Network Analysis

If the network is reasonably linear, PRN and random noise both give the same results as the swept-sine test of other analyzers. But PRN gives the frequency response much faster. PRN can be used to measure the frequency response in a single time record. Because the random source is true noise, it must be averaged for several time records before an accurate frequency response can be determined. Therefore, PRN is the best stimulus to use with fairly linear networks because it gives the fastest results.*

Non-Linear Network Analysis

If the network is severely non-linear, the situation is quite different. In this case, PRN is a very poor test signal and random noise is much better. To see why, let us look at just two of the sine waves that compose the PRN source. We see in Figure 6.4 that if two sine waves are put through a nonlinear network, distortion products will be generated equally spaced from the signals.** Unfortunately, these products will fall exactly on the frequencies of the other sine waves in the PRN. So the distortion products add to the output and therefore interfere with the measurement of the frequency response. Figure 6.5a shows the jagged response of a nonlinear network measured with PRN. Because the PRN source repeats itself exactly every time record, this noisy-looking trace never changes and will not average to the desired frequency response.

With random noise, the distortion components are also random and will average out. Therefore, the frequency response does not include the distortion and we get the more reasonable results shown in Figure 6.5b.

This points out a fundamental problem with measuring non-linear networks; *the frequency response is not a property of the network alone, it also depends on the stimulus.* Each stimulus, swept-sine, PRN and random noise will, in general, give a different result. Also, if the amplitude of the stimulus is changed, you will get a different result.

* There is another reason why PRN is a better test signal than random or linear networks. Recall from the last section that PRN is self-windowing. This means that unlike random noise, pseudo-random noise has no leakage. Therefore, with PRN, we can measure lightly damped (high Q) resonances more easily than with random noise.

** This distortion is called intermodulation distortion.

To illustrate this, consider the mass-spring system with stops that we used in Agilent Application Note 1405-1. If the mass does not hit the stops, the system is linear and the frequency response is given by Figure 6.6a.

If the mass does hit the stops, the output is clipped and a large number of distortion components are generated. As the output approaches a square wave, the fundamental component becomes constant. Therefore, as we increase the input amplitude, the gain of the network drops. We get a frequency response like Figure 6.6b, where the gain is dependent on the input signal amplitude.

So, as we have seen, the frequency response of a nonlinear network is not well defined, i.e., it depends on the stimulus. Yet it is often used in spite of this. The frequency response of linear networks has proven to be a very powerful too, and so naturally people have tried to extend it to non-linear analysis, particularly since other nonlinear analysis tools have proved intractable.

If every stimulus yields a different frequency response, which one should we use? The "best" stimulus could be considered to be one that approximates the kind of signals you would expect to have as normal inputs to the network. Since any large collection of signals begins to look like noise, noise is a good test signal.* As we have already explained, noise is also a good test signal because it speeds the analysis by exciting all the filters of our analyzer simultaneously.

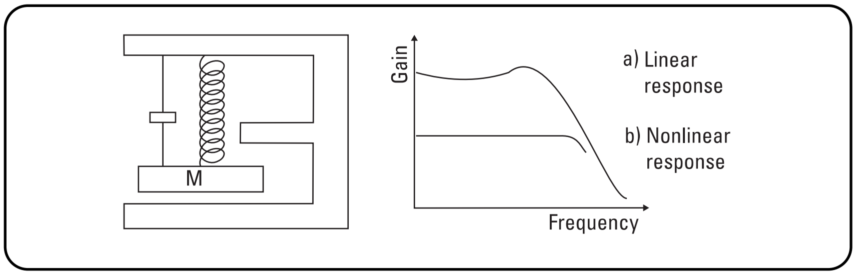


Figure 6.6. Nonlinear system

But many other test signals can be used with dynamic signal analyzers and are "best" (optimum) in other senses. As explained in the beginning of this section, sine waves can be used to give the same results as other types of network analyzers although the speed advantage of the dynamic signal analyzer is lost. A fast sine sweep (chirp) will give very similar results with all the speed of dynamic signal analysis, and so is a better test signal. An impulse is a good test signal for acoustical testing if the network is linear. It is good for acoustics because reflections from surfaces at different distances can easily be isolated or eliminated if desired. For instance, by using the "force" window described earlier, it is easy to get the free field response of a speaker by eliminating the room reflections from the windowed time record.

Band-Limited Noise

Before leaving the subject of network stimulus, it is appropriate to discuss the need to band limit the stimulus. We want all the power of the stimulus to be concentrated in the frequency region we are analyzing. Any power outside this region does not contribute to the measurement and could excite non-linearities. This can be a particularly severe problem when testing with random noise since it theoretically has the same power at all frequencies (white noise). To eliminate this problem, dynamic signal analyzers often limit the frequency range of their built-in noise stimulus to the frequency span selected. This could be done with an external noise source and filters, but every time the analyzer span changed, the noise power and filter would have to be readjusted. This is done automatically with a built-in noise source so transfer function measurements are easier and faster.

* This is a consequence of the central limit theorem. As an example, the telephone companies have found that when many conversations are transmitted together, the result is like white noise. The same effect is found more commonly at a crowded cocktail party.

Section 7: Averaging

To make it as easy as possible to develop an understanding of dynamic signal analyzers we have almost exclusively used examples with deterministic signals, i.e., signals with no noise. However, the real world is rarely so obliging. The desired signal often must be measured in the presence of significant noise. At other times the “signals” we are trying to measure are more like noise themselves. Common examples that are somewhat noise-like include speech, music, digital data, seismic data and mechanical vibrations. Because of these two common conditions, we must develop techniques both to measure signals in the presence of noise and to measure the noise itself.

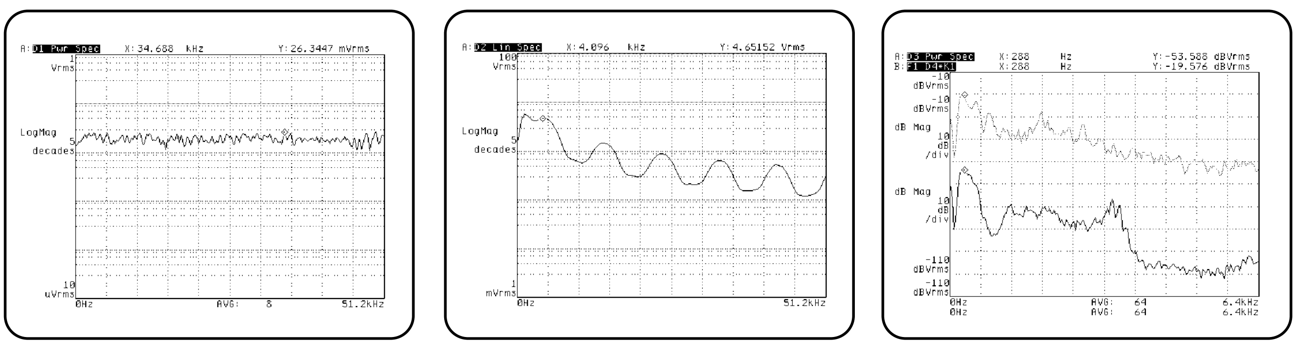
The standard technique in statistics to improve the estimates of a value is to average. When we watch a noisy reading on a dynamic signal analyzer, we can guess the average value. But because the dynamic signal analyzer contains digital computation capability, we can have it compute this average value for us. Two kinds of averaging are available, RMS (or “power” averaging) and linear averaging.

RMS Averaging

When we watch the magnitude of the spectrum and attempt to guess the average value of the spectrum component, we are doing a crude RMS* average. We are trying to determine the average magnitude of the signal, ignoring any phase difference that may exist between the spectra. This averaging technique is very valuable for determining the average power in any of the filters of our dynamic signal analyzers. The more averages we take, the better our estimate of the power level.

In Figure 7.1, we show RMS averaged spectra of random noise, digital data and human voices. Each of these examples is a fairly random process, but when averaged we can see the basic properties of its spectrum.

If we want to measure a small signal in the presence of noise, RMS averaging will give us a good estimate of the signal plus noise. We can not improve the signal-to-noise ratio with RMS averaging; we can only make more accurate estimates of the total signal-plus-noise power.



a) Random noise

b) Digital data

c) Voices. Traces were separated 30 dB for clarity
Upper trace: female speaker
Lower trace: male speaker

Figure 7.1. RMS averaged spectra

* RMS stands for “root-mean-square” and is calculated by squaring all the values, adding the squares together, dividing by the number of measurements (mean) and taking the square root of the result.

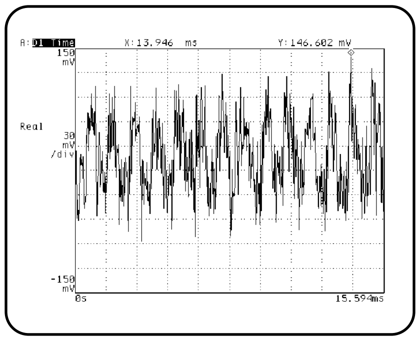
Linear Averaging

However, there is a technique for improving the signal-to-noise ratio of a measurement, called *linear averaging*. It can be used if a trigger signal is available that is synchronous with the periodic part of the spectrum. Of course, the need for a synchronizing signal is somewhat restrictive, although there are numerous situations in which one is available. In network analysis problems, the stimulus signal itself can often be used as a synchronizing signal.

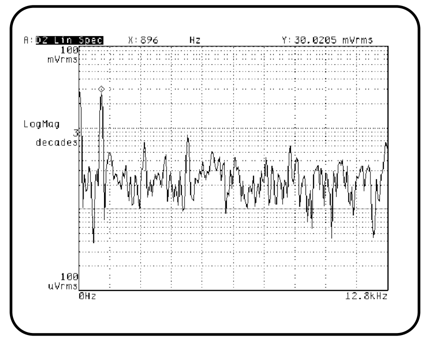
Linear averaging can be implemented many ways, but perhaps the easiest to understand is where the averaging is done in the time domain. In this case, the synchronizing signal is used to trigger the start of a time record. Therefore, the periodic part of the input will always be exactly the same in each time record we take, whereas the noise will, of course, vary. If we add together a series of these triggered time records and divide by the number of records we have taken, we will compute what we call a linear average.

Since the periodic signal will have repeated itself exactly in each time record, it will average to its exact value. But since the noise is different in each time record, it will tend to average to zero. The more averages we take, the closer the noise comes to zero and we continue to improve the signal-to-noise ratio of our measurement.

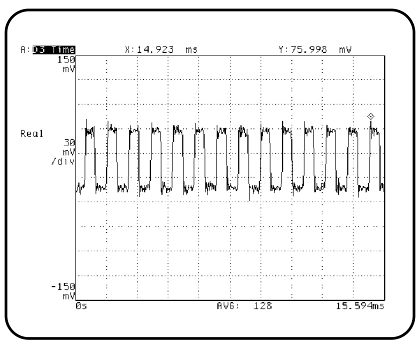
Figure 7.2 shows a time record of a square wave buried in noise. The resulting time record after 128 averages shows a marked improvement in the signal to noise ratio. Transforming both results to the frequency domain shows how many of the harmonics can now be accurately measured because of the reduced noise floor.



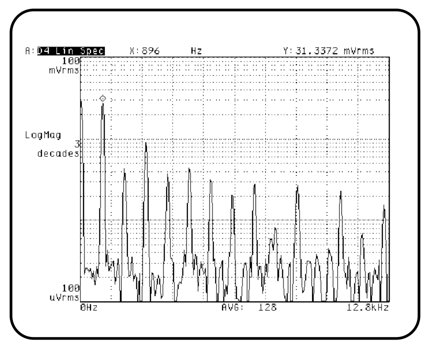
a) Single record, no averaging



b) Single record, no averaging



c) 128 linear averages



d) 128 linear averages

Figure 7.2. Linear averaging

Section 8: Real-Time Bandwidth

Until now we have ignored the fact that it will take a finite time to compute the FFT of our time record. In fact, if we could compute the transform in less time than our sampling period we could continue to ignore this computational time. Figure 8.1 shows that under this condition we could get a new frequency spectrum with every sample. As we have seen from the section on aliasing, this could result in far more spectrums every second than we could possibly comprehend. Worse, because of the complexity of the FFT algorithm, it would take a very fast and very expensive computer to generate spectrums this rapidly.

A reasonable alternative is to add a time record buffer to the block diagram of our analyzer (Figure 8.2). In Figure 8.3 we can see that this allows us to compute the frequency spectrum of the previous time record while gathering the current time record. If we can compute the transform before the time record buffer fills, then we are said to be *operating in real time*.

To see what this means, let us look at the case where the FFT computation takes longer than the time to fill the time record. The case is illustrated in Figure 8.4. Although the buffer is full, we have not finished the last transform, so we will have to stop taking data. When the transform is finished, we can transfer the time record to the FFT and begin to take another time record. This means that we missed some input data and so we are said to be *not operating in real time*.

Recall that the time record is not constant but deliberately varied to change the frequency span of the analyzer. For wide frequency spans, the time record is shorter. Therefore, as we increase the frequency span of the analyzer, we eventually reach a span where the time record is equal to the FFT computation time. This frequency span is called the *real-time bandwidth*. For frequency spans at and below the real-time bandwidth, the analyzer does not miss any data.

Real-Time Bandwidth Requirements

How wide a real-time bandwidth is needed in a dynamic signal analyzer? Let us examine a few typical measurements to get a feeling for the considerations involved.

Adjusting Devices

If we are measuring the spectrum or frequency response of a device that we are adjusting, we need to watch the spectrum change in what might be called *psychological real time*. A new spectrum every few tenths of a second is sufficiently fast to allow an operator to watch adjustments *in what he or she would consider to be real time*. However, if the response time of the device under test is long, the speed of the analyzer is immaterial. We will have to wait for the device to respond to the changes before the spectrum will be valid, *no matter how many spectrums we generate in that time*. This is what makes adjusting lightly damped (high Q) resonances tedious.

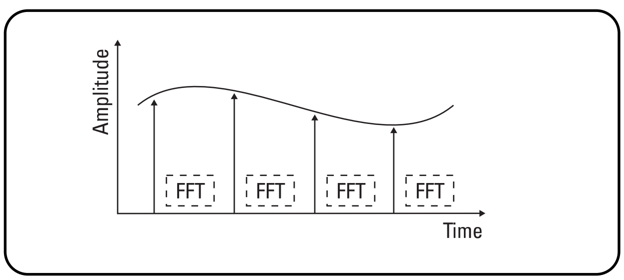


Figure 8.1. A new transform every sample

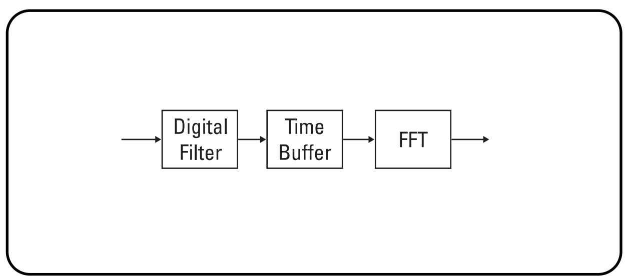


Figure 8.2. Time buffer added to block diagram

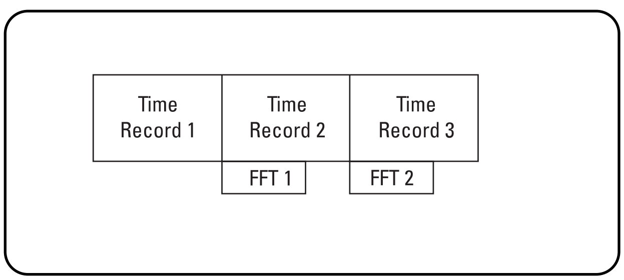


Figure 8.3. Real-time operation

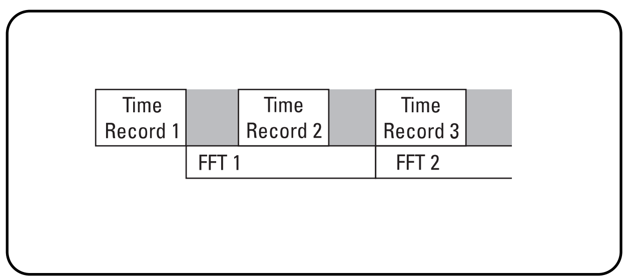


Figure 8.4. Non-real-time operation

RMS Averaging

A second case of interest in determining real-time bandwidth requirements is measurements that require RMS averaging. We might be interested in determining the spectrum distribution of the noise itself or in reducing the variation of a signal contaminated by noise. There is no requirement in averaging that the records must be consecutive with no gaps. Therefore, a small real-time bandwidth will not affect the accuracy of the results.

However, the real time bandwidth will affect the speed with which an RMS averaged measurement can be made. Figure 8.5 shows that for frequency spans above the real-time bandwidth, the time to complete the average of N records is dependent only on the time to compute the N transforms. Rather than continually reducing the time to compute the RMS average as we increase our span, we reach a fixed time to compute N averages.

Therefore, a small real-time bandwidth is only a problem in RMS averaging when large spans are used with a large number of averages. Under these conditions we must wait longer for the answer. Since wider real-time bandwidths require faster computations and therefore a more expensive processor, there is a straight-forward trade-off of time versus money. In the case of RMS averaging, higher real-time bandwidth gives you somewhat faster measurements at increased analyzer cost.

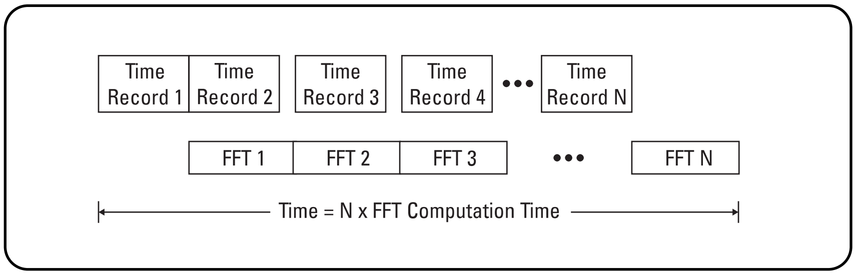


Figure 8.5. RMS averaging time

Transients

The last case of interest in determining the needed real-time bandwidth is the analysis of transient events. If the entire transient fits within the time record, the FFT computation time is of little interest. The analyzer can be triggered by the transient and the event stored in the time record buffer. The time to compute its spectrum is not important.

However, if a transient event contains high-frequency energy and lasts longer than the time record necessary to measure the high-frequency energy, then the processing speed of the analyzer is critical. As shown in Figure 8.6b, some of the transient will not be analyzed if the computation time exceeds the time record length.

In the case of transients longer than the time record, it is also imperative that there is some way to rapidly record the spectrum. Otherwise, the information will be lost as the analyzer updates the display with the spectrum of the latest time record. A special

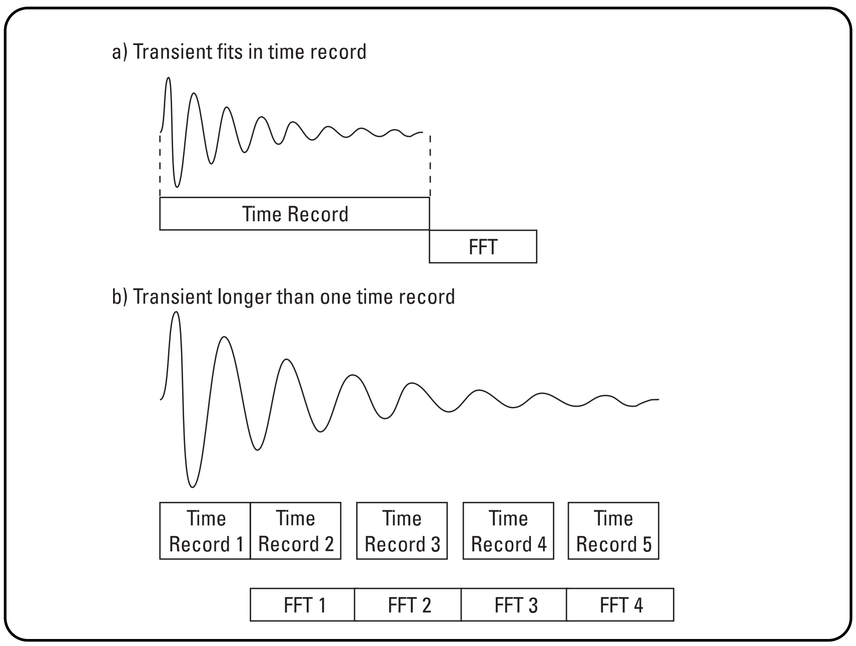


Figure 8.6. Transient analysis

display which can show more than one spectrum (“waterfall” display), mass memory, a high-speed link to a computer or a high-speed facsimile recorder is needed. The output device must be able to record a spectrum every time record or information will be lost. Fortunately, there is an easy way to avoid the need for an expensive wide real-time bandwidth analyzer and an expensive, fast spectrum recorder. One-time transient events like explosions and pass-by noise are usually digitally recorded for later analysis because of the expense of repeating the test. Continuously sampled time data can be recorded into a large time-capture memory or a high-speed through-put disk. This allows you to analyze the data later with no information loss.

So we see that there is no clear-cut answer to what real-time bandwidth is necessary in a dynamic signal analyzer. Except in analyzing long transient events, the added expense of a wide real-time bandwidth gives little advantage. It is possible to analyze long transient events with a narrow real-time bandwidth analyzer, but it does require the recording of the input signal. This method is slow and requires some operator care, but you can avoid purchasing an expensive analyzer and fast spectrum recorder. It is a clear case of speed of analysis versus dollars of capital equipment.

Section 9: Overlap Processing

In Section 8 we considered the case where the computation of the FFT took longer than the time it took to collect the time record. In this section we will look at a technique, overlap processing, which can be used when the FFT computation takes less time than gathering the time record.

To understand overlap processing, let us look at Figure 9.1a. We see a low-frequency analysis where gathering the time record takes much longer than the FFT computation time. Our FFT processor is sitting idle much of the time. If instead of waiting for an entirely new time record we *overlapped* the new time record with some of the old data, we would get a new spectrum as often as we computed the FFT. This *overlap processing* is illustrated in Figure 9.1b. To understand the benefits of overlap processing, let us look at the same cases we used in the last section.

Adjusting Devices

We saw in the last section that we need a new spectrum every few tenths of a second when adjusting devices. Without overlap processing this limits our resolution to a few Hertz. With overlap processing our resolution is unlimited. But we are not getting something for nothing. Because our overlapped time record contains old data from before the device adjustment, it is not completely correct. It does indicate the direction and the amount of change, but we must wait a full time record after the change for the new spectrum to be accurately displayed.

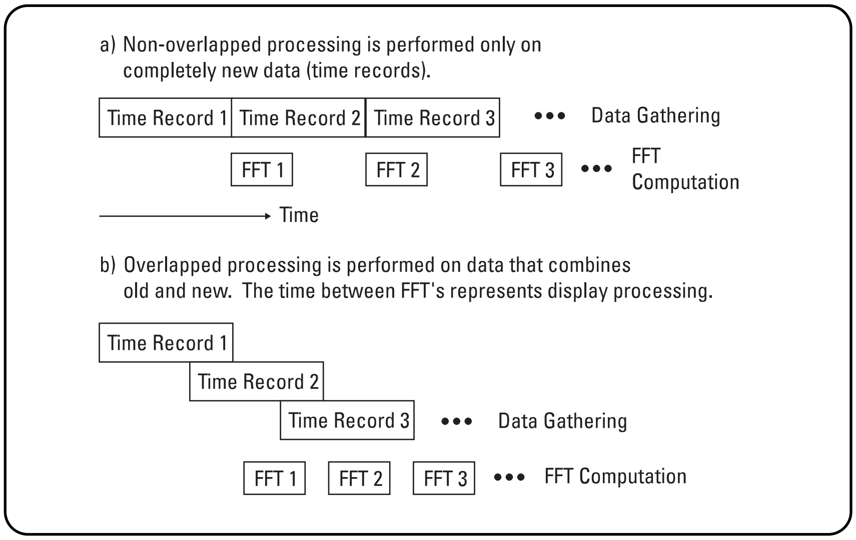


Figure 9.1. Understanding overlap processing

Nonetheless, by indicating the direction and magnitude of the changes every few tenths of a second, overlap processing does help in the adjustment of devices.

RMS Averaging

Overlap processing can give dramatic reductions in the time to compute RMS averages with a given variance. Recall that window functions reduce the effects of leakage by weighting the ends of the time record to zero. Overlapping eliminates most or all of the time that would be wasted taking this data. Because some overlapped data is used twice, more averages must be taken to get a given variance than in the non-overlapped case. Figure 9.2 shows the improvements that can be expected by overlapping.

Transients

For transients shorter than the time record, overlap processing is useless. For transients longer than the time record, the real-time bandwidth of the analyzer and spectrum recorder is usually a limitation. If it is not, overlap processing allows more spectra to be generated from the transient, usually improving resolution of resulting plots.

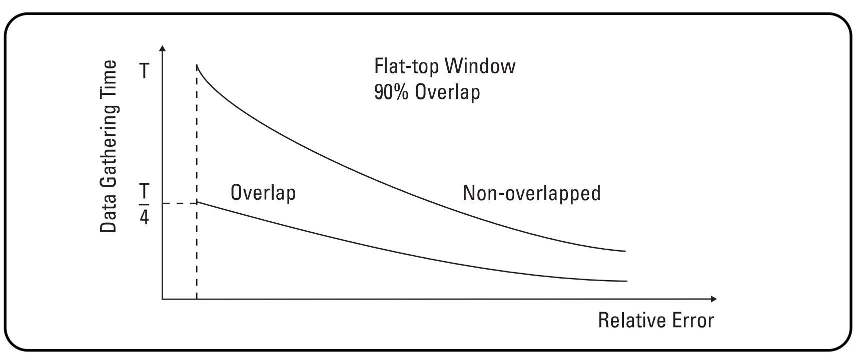


Figure 9.2. RMS averaging speed improvements with overlap processing

Summary

In this application note, we have developed the basic properties of dynamic signal analyzers. We found that many properties could be understood by considering what happens when we transform a finite, sampled time record. The length of this record determines how closely our filters can be spaced in the frequency domain and the number of samples determines the number of filters in the frequency domain. We also found that unless we filtered the input we could have errors due to aliasing, and that finite time records could cause a problem called *leakage* that we minimized by windowing. We then added several features to our basic dynamic signal analyzer to enhance its capabilities. Band-selectable analysis allows us to make high-resolution measurements even at high frequencies. Averaging gives more accurate measurements when noise is present and even allows us to improve the signal-to-noise ratio when we can use linear averaging. Finally, we incorporated a noise source in our analyzer to act as a stimulus for transfer function measurements.

Bibliography

Bendat, Julius S. and Piersol, Allan G., *"Random Data: Analysis and Measurement Procedures,"* Wiley-Interscience, New York, 1971.

Bendat, Julius S. and Piersol, Allan G., *"Engineering Applications of Correlation and Spectral Analysis,"* Wiley-lnterscience, New York, 1980.

Bracewell, R., *"The Fourier Transform and its Applications,"* McGraw-Hill, 1965.

Cooley, J.W. and Tukey, J.W., *"An Algorithm for the Machine Calculation of Complex Fourier Series,"* Mathematics of Computation, Vo. 19, No. 90, p. 297, April 1965.

McKinney, W., *"Band Selectable Fourier Analysis,"* Hewlett-Packard Journal, April 1975, pp. 20-24.

Otnes, R.K. and Enochson, L., *"Digital Time Series Analysis,"* John Wiley, 1972.

Potter, R. and Lortscher, J., *"What in the World is Overlap Processing,"* Hewlett-Packard Santa Clara DSA/Laser Division *"Update,"* Sept. 1978.

Ramse , K.A., *"Effective Measurements for Structural Dynamics Testing,"* Part 1, Sound and Vibration Magazine, November 1975, pp. 24-35.

Roth, P., *"Effective Measurements Using Digital Signal Analysis,"* IEEE Spectrum, April 1971, pp. 62-70.

Roth, P., *"Digital Fourier Analysis,"* Hewlett-Packard Journal, June 1970.

Welch, Peter D., *"The Use of Fast Fourier Transform for the Estimation of Power Spectra: A Method Based on Time Averaging Over Short, Modified Periodograms,"* IEEE Transactions on Audio and Electro-acoustics, Vol. AU-15, No. 2, June 1967, pp. 70-73.

Related Agilent Literature

Agilent Application Note – *Introduction to Time, Frequency and Modal Domains*, pub. no. 1405-1

Agilent Application Note – *Using Dynamic Signal Analysers*, pub. no. 1405-3

Agilent Application Note – *The Fourier Transform: A Mathematical Background*, pub. no. 1405-4

Product Overview – *Agilent 35670A Dynamic Signal Analyzer,* pub. no. 5966-3063E

Product Overview – *Agilent E1432/33/34 VXI Digitizers/Source,* pub. no. 5968-7086E

Product Overview – *Agilent E9801B Data Recorder/Logger,* pub. no. 5968-6132E

Glossary

Aliasing — a phenomenon that can occur when a signal is not sampled at greater than twice the maximum frequency component; high-frequency signals appear as low-frequency components; avoided by filtering out signals greater than 1/2 the sample rate

Anti-alias filter — a low pass filter installed before the sampler and analog-to-digital converter to limit the input frequency range of a signal to prevent aliasing; designed to filter out frequencies greater than 1/2 the sample rate (typically 1/2.56 to allow for filter rolloff)

Band-selectable analysis — an analysis capability that allows you to "zoom in" for a high-resolution close-up shot of the frequency spectrum by concentrating filters in the frequency range of interest.

Digital filter — a decimating filter that filters the digital representation of the input signal (after it has been sampled and digitized) to the desired frequency span. It also reduces the sample rate at its output to the rate needed for that frequency span.

Fast Fourier Transform (FFT) — an algorithm used in computers and DSAs to compute discrete frequency components from sampled time data; invented by Cooley and Tukey

Flat-top window — a windowing function that minimizes amplitude error for off-center input-signal components

Force window — a windowing function that eliminates stray signals; used for the excitation signal in impact test to improve signal-to-noise ratio

Hanning window — a windowing function used to reduce leakage when measuring noise and periodic signals like sine waves

Leakage — the spreading of energy throughout the frequency domain; energy leaks out of one resolution line of an FFT into other lines, sometimes masking small signals close to a sine wave. This happens when the signal is not periodic within a time record. Applying an appropriate window function will minimize the amplitude error due to the leakage.

Linear averaging — a technique for improving the signal-to-noise ratio of a measurement; can be used if a trigger signal is available that is synchronous with the periodic part of the spectrum

Lines — To reduce confusion about which domain we are in, samples in the frequency domain are called lines.

Nyquist Criterion — the minimum theoretical sample rate for a baseband signal to be reproduced in sampled form, equal to twice the highest frequency of the input signal. In most cases, you will want to use a higher sample rate to represent the signal accurately.

Periodic random noise (PRN) — a kind of pseudo-random noise that periodically repeats its sequence; the best stimulus for testing linear networks.

Pseudo-random noise — a mathematically generated random noise whose period is matched to time record length, thus eliminating leakage

Random noise — true noise

Real time — If we can compute the transform (FFT) before the time record buffer fills, we are said to be operating in real time.

Rectangular window — another term for uniform window

Response window — an exponential-weighted window that guarantees that the response dies out (goes to zero) within the time record; used in impact test to avoid leakage error; also called "exponential window"

RMS averaging — a technique for measuring small signals in the presence of noise. RMS averaging is useful for processing stationary signals. Each data block can be overlapped to achieve the maximum number of averages. RMS averaging produces amplitude information only. You lose phase information.

Self-windowing functions — a function that does not require a window because it occurs completely within the time record or its period is matched to time-record length (it generates no leakage in the FFT)

Time record — a block of N consecutive, equally spaced samples of the input; this block is the basic unit transformed by an FFT.

Transfer function — a ratio of the output over the input, both in the Laplace domain; sometimes used interchangeably with "frequency response function"

Uniform window — a windowing function that weights all of the time record uniformly; used for transient signals

Windowing — a way to reduce leakage by forcing an FFT to look at data through a narrow window where the input is zero at both ends of the time record. Many different functions can be used to window the data, depending on the type of measurement you are making.

Zoom — another term for band-selectable analysis

www.agilent.com

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

By internet, phone, or fax, get assistance with all your test & measurement needs
Online assistance:
www.agilent.com/find/assist

Phone or Fax

United States:
(tel) 1 800 452 4844

Canada:
(tel) 1 877 894 4414
(fax) (905) 282 6495

China:
(tel) 800 810 0189
(fax) 800 820 2816

Europe:
(tel) (31 20) 547 2323
(fax) (31 20) 547 2390

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Korea:
(tel) (82 2) 2004 5004
(fax) (82 2) 2004 5115

Latin America:
(tel) (305) 269 7500
(fax) (305) 269 7599

Taiwan:
(tel) 0800 047 866
(fax) 0800 286 331

Other Asia Pacific Countries:
(tel) (65) 6375 8100
(fax) (65) 6836 0252
(e-mail) tm_asia@agilent.com

Product specifications and descriptions in this document subject to change without notice.
© Agilent Technologies, Inc. 1994, 1995, 1999, 2000, 2002
Printed in the USA August 7, 2002
5988-6774EN

Agilent Technologies