



ME'scopeVES Application Note #2

Waveform Integration & Differentiation

INTRODUCTION

The most common type of vibration transducer is an accelerometer, which measures acceleration. Therefore, in order to answer the question "How much is the machine or structure really moving?" a common requirement of signal processing is to integrate acceleration (or velocity) signals to displacements.

In this note, we will exercise both the integration and differentiation methods in ME'scopeVES. Integration and differentiation can be done on either time domain or frequency domain waveforms. We will see how **DC offsets and leakage** can cause errors when integrating waveforms, and how these errors can be dealt with effectively.

Since ME'scopeVES has a built-in FFT, any waveform can be easily transformed from one domain to the other. Therefore, either time or frequency domain integration or differentiation can be used on any waveform.

TIME DOMAIN INTEGRATION

The main difficulty with the repeated use of time domain integration (for example, double integration to obtain displacement from acceleration) is that any DC offset **must be removed** before integration is performed. Otherwise, the integrated DC offset will dominate the result.

Time domain waveforms are integrated by using the **trapezoidal rule**. That is, the area under the curve is approximated with a summation,

$$\int \mathbf{x}(t)dt \cong \sum_{i=1}^N \left[\frac{\mathbf{x}_i + \mathbf{x}_{i-1}}{2} \right] \Delta t$$

where:

$\mathbf{x}(t)$ = continuous time domain waveform.

\mathbf{x}_i = i^{th} sample of the time waveform.

$\mathbf{x}_0 = 0$

Δt = time increment between samples.

N = number of samples (block size)

FREQUENCY DOMAIN INTEGRATION

Frequency domain integration is done by dividing the waveform by frequency values. This, by itself, is accurate.

However, the main difficulty with the frequency domain method is that if you want to integrate a time domain waveform, you must first FFT it to the frequency domain. If the waveform is not **periodic in the window** (see Application Note #1 for details), **leakage** will occur, which can cause significant errors in the results.

Frequency domain waveforms are integrated by using the following **equivalent** frequency domain operation,

$$\int \mathbf{x}(t)dt \Leftrightarrow \frac{\mathbf{X}_i(2\pi\mathbf{f}_i)}{(j2\pi\mathbf{f}_i)}, \quad i=1, \dots, N/2$$

where:

$\mathbf{X}_i(2\pi\mathbf{f}_i)$ = linear spectrum (FFT) of the signal for the i^{th} sample.

$(j2\pi\mathbf{f}_i)$ = frequency of the i^{th} sample in rad./sec.

\mathbf{f}_i = frequency of the i^{th} sample in Hz.

j - denotes the imaginary operator.

In other words, integration of a time domain waveform is equivalent to **dividing** each sample of its frequency spectrum ($\mathbf{X}_i(2\pi\mathbf{f}_i)$) by the frequency ($j2\pi\mathbf{f}_i$).

TIME DOMAIN DIFFERENTIATION

When the **Tools | Math | Differentiate** command is executed on time domain waveforms, the data is differentiated using the following rules,

- The first sample is differentiated using a **forward difference** formula.
- The last sample is differentiated using a **backward difference** formula.
- All other samples are differentiated using a **central difference** formula.

FREQUENCY DOMAIN DIFFERENTIATION

Differentiation is performed on frequency domain data using the following **equivalent** operation,

$$\frac{d(\mathbf{x}(t))}{dt} \Leftrightarrow (j2\pi\mathbf{f}_i)(\mathbf{X}_i(2\pi\mathbf{f}_i)), \quad i=1, \dots, N/2$$

In other words, differentiation of a time domain waveform is equivalent to **multiplying** each sample of its frequency spectrum ($X_i(2\pi f_i)$) by the frequency of the sample ($j2\pi f_i$).

PERIODIC SIGNALS

If a waveform is **periodic in the sampling window**, we will see that it can be accurately integrated and differentiated **in either domain**. To demonstrate this, use ME’scopeVES to synthesize a sine wave that is periodic in the window.

- Execute **File | New | Data Block** in the ME’scopeVES window. Name the Data Block file **3125.BLK**, and click **OK**. The **Synthesize Time Traces** dialog box will open.

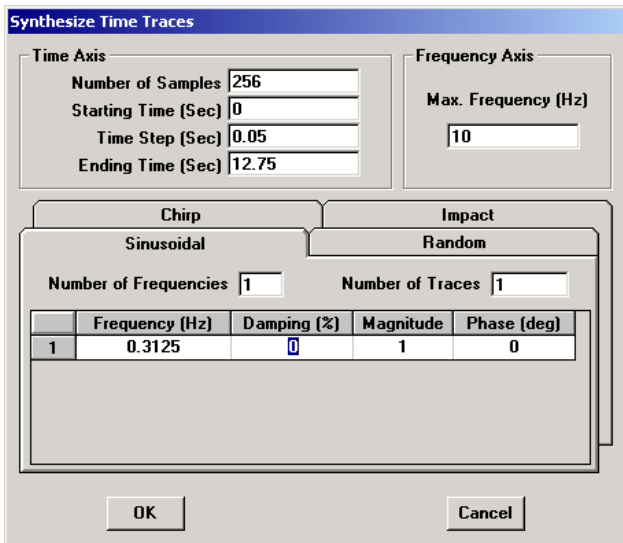


Figure 1. Dialog Box for Periodic Sine.

- Enter the following numbers into the dialog box,

Number of Samples: 256
Max. Frequency: 10 Hz
Number of Frequencies: 1
Number of Traces: 1
Frequency: 0.3125, Damping (%): 0
Magnitude: 1, Phase (deg): 0

These parameters will be used to synthesize a **0.3125 Hz** sine wave with a **magnitude** of **1.0** and no damping.

- When all of the parameters are entered, press the **OK** button to synthesize the sine wave. The new Data Block window will open with the sine wave in it.
- Execute **File | Save** and save the file to the disk.

Notice that the Trace has **exactly 4 cycles** of the sine wave in it. This signal is **periodic in the window**, since an integer number of cycles have been sampled (in this case synthesized) within the sampling window.

Notice also that the sine wave units are **Gs** (gravitational units). The Integration commands in ME’scopeVES will convert the units from acceleration to velocity, and velocity to displacement as you integrate signals.

NOTE: G’s are automatically converted to (meters per second-squared) by the ME’scopeVES integration commands.

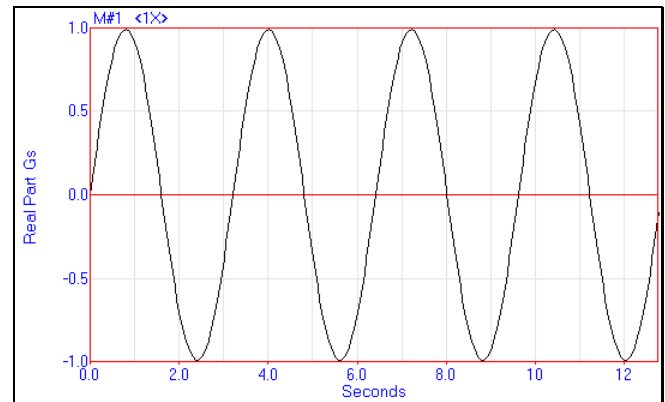


Figure 2. Periodic Acceleration Sine Wave.

Time Domain Integration of a Periodic Signal

To integrate the signal shown in Figure2,

- Execute **Tools | Math | Integrate** in the Data Block window.

After the integration takes place, the acceleration signal will be replaced with the velocity signal shown in Figure 3. Notice that the units of the signal are **meters/second**.

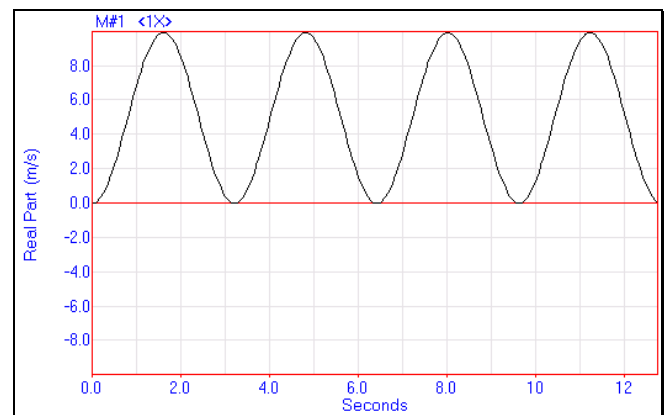


Figure 3. Velocity After Integration.

Notice that the integrated signal in Figure 3 has a large bias (or DC offset) in it. That is, instead of being a cosine wave ranging between (+) and (-) the same value, it ranges between **0** and **9.98**. This corresponds to the following integral.

$$v(t) = \int_{\tau=0}^t a(\tau) d\tau$$

$$v(t) = 9.80665 \frac{\text{m}}{\text{s}^2} \int_{\tau=0}^t \sin(.3125 * 2\pi\tau) d\tau$$

$$v(t) = \frac{9.80665 \frac{\text{m}}{\text{s}^2}}{.3125 * 2\pi \frac{\text{rad}}{\text{s}}} (-\cos(.3125 * 2\pi t) + \cos(0))$$

$$v(t) = 4.99449 \frac{\text{m}}{\text{s}} (1 - \cos(.3125 * 2\pi t))$$

Setting $t=0$ produces $v(0)=0$. At $t=1.6$ sec, $v(1.6)=9.9898$ m/s.

NOTE: The accuracy of the integrated signal increases as Δt decreases. With a $\Delta t = 0.00625$, integration yields a $v(1.6)=9.9888$ m/s.

DC REMOVAL

If we integrated $v(t)$ a second time, the 4.9949m/s constant would create a ramp function in the data. In more general terms, *integration greatly amplifies the low frequencies* in a waveform, including DC (zero frequency). This is made clear by the frequency domain integration equation. Dividing each sample of a waveform's spectrum by its frequency is the same as multiplying it by the function (1/Frequency). *Double integration multiples it by (1/Frequency)².*

Unfortunately, most real world signals have *some amount* of DC offset in them, even when DC coupling is used in the acquisition system to remove it. Even a small amount of DC will *dominate* the result when integration is performed.

Time domain integration is the more straightforward method to use on time domain waveforms, if they don't have any DC offsets in them. Unfortunately, there is no foolproof method for removing a DC offset directly from a time domain signal, unless it is a cyclic waveform.

Time Domain DC Removal

The **Tools | Math | Remove DC** command in ME'scopeVES removes DC from time domain waveforms. It allows you to choose between two methods for computing the DC value.

$$\text{Method \#1: DC} = \frac{(\text{Min} + \text{Max})}{2}$$

where:

Min = the minimum waveform value.

Max = the maximum waveform value.

The average of minimum & maximum values works well for noise-free cyclic signals, such as sine, triangle, and square waves. On the other hand, it can give invalid results for signals such as transients, pulses, etc.

Method #2: DC = Average Value of the Trace

Both methods work correctly for signals that are periodic in the sampling window. The average Trace value method will give invalid results for signals that are non-periodic in the window.

Using the Cursor Band

If the **Cursor Band** is turned ON, both methods will compute the DC value by using only the Trace data in the band. You can usually improve your results by using the band cursor to limit the data used by this command.

To remove the DC offset from the velocity signal shown in Figure 3,

- Execute **Tools | Math | Remove DC**. Select **No** to compute DC as the average of minimum & maximum values, or **Yes** to compute DC as the average of the Trace values.

Notice that the velocity waveform now ranges between **+4.99** and **-4.99**. Now, it can be integrated again to obtain displacement.

- Execute **Tools | Math | Integrate** again.
- Execute **Tools | Math | Remove DC** again.

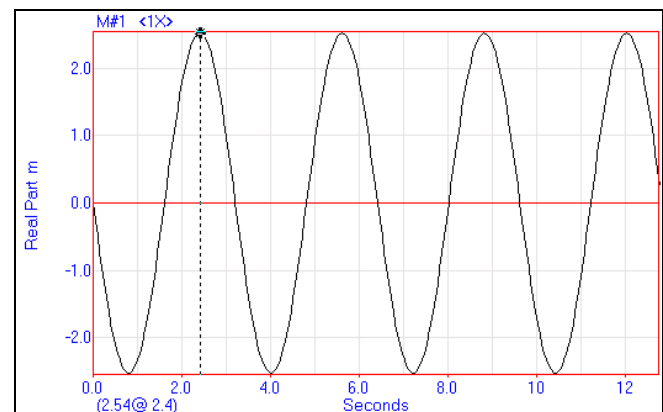


Figure 4. Displacement After Double Integration.

The resulting displacement (in meters), is shown in Figure 4. Notice (from the cursor value) that its ranges between **+2.54** and **-2.54** meters, the expected result.

TIME DOMAIN DIFFERENTIATION

Now, let's double differentiate the displacement time waveform in Figure 4 to recover the original acceleration signal.

- Execute **Tools | Math | Differentiate twice**.
- Execute **Tools | Math | Remove DC**.

Changing Units

The resulting acceleration signal is shown in Figure 5. Notice that its values range between **+9.76** and **-9.76 m/s²**. These are approximately **+1.0** and **-1.0 Gs**.

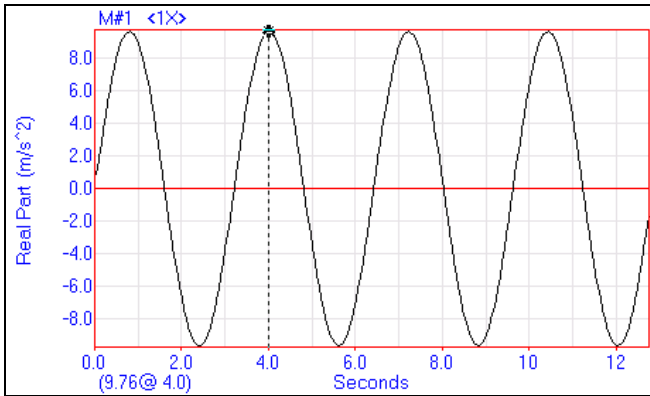


Figure 5. Acceleration After Double Differentiation.

To change units from **m/s²** to **Gs**,

- Drag the **Vertical Blue Bar** in the Data Block window to the left, to expose the Traces spreadsheet.
- Double click on the **Units** column heading. A dialog box will open. Select **Gs** from the drop down list, and click on **OK**.

Now, the original acceleration signal with **unit amplitude** and units of **Gs** should be displayed.

FREQUENCY DOMAIN INTEGRATION

So far, we have seen that time domain integration can be performed repeatedly on time domain signals, but DC removal must be performed before integrating each time.

To double integrate the same periodic sine wave shown in Figure 2, in the frequency domain,

- Execute **File | Add** to re-open the Data Block file **3125.BLK** from the disk.

- Execute **Transform | FFT** to obtain the Linear spectrum of the sine wave.
- Execute **Tools | Math | Integrate twice**.
- Execute **Transform | Inverse FFT** to obtain the displacement sine wave.

The same signal as that shown in Figure 4 should be displayed. Again, the units of the signal are meters, and that the maximum amplitude is **2.54** meters.

To recover the original signal, try transforming to the frequency domain, double differentiating, transforming back to the time domain, and changing units back to **Gs**, as done before.

So far, we have seen that both time and frequency integration & differentiation yield the same result when applied to a sine wave that is *periodic in the sampling window*. Now, let's look at a case where DC Removal doesn't work properly.

TIME VS. FREQUENCY DOMAIN DC REMOVAL

The **Tools | Math | Remove DC** command in ME'scopeVES only operates on time domain Traces. We have already seen that it works properly on cyclic data such as sine waves. On the other hand, DC offsets can always be removed in the frequency domain by using the band pass window to zero the spectrum values at or near DC.

In the next example, we will see that DC removal works in the frequency domain, but not in the time domain.

To use the DC Removal command first,

- Execute **File | Add** and select **PLTMODES.BLK** file from the Examples directory.
- Execute **Format | Vertical Axis Scaling** and select **Linear** magnitude. **Zoom** around DC (zero frequency), and scroll through the measurements.

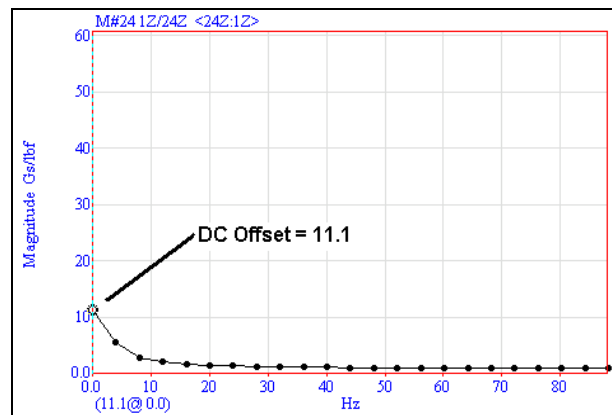


Figure 6. M# 24 Showing DC Offset.

Notice that all of the measurements have a non zero value at DC (the first sample in each Trace). The DC offset of **M# 24** is shown in Figure 6. When transformed to the time domain, all of these measurements will have DC offsets in them.

Now, transform all of the FRFs to Impulse Response Functions (IRFs) in the time domain,

- Execute **Transform | Inverse FFT**.

When the Inverse FFT has completed, the Impulse Responses will be displayed, as shown in Figure 7.

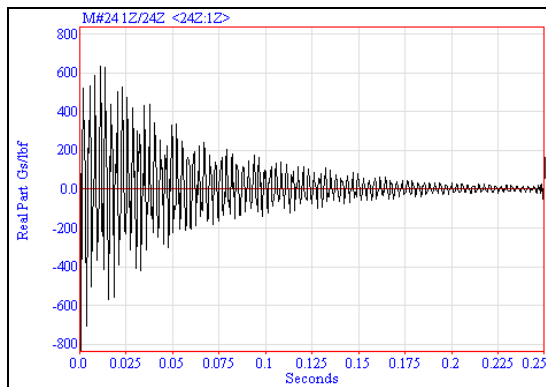


Figure 7. IRF of M# 24.

Time Domain DC Removal

First, we will try to remove DC from the impulse responses by using the time domain method.

- Execute **Tools | Math | Remove DC**. A dialog box will open. Select **No** to compute DC as the average of the minimum & maximum values.
- Execute **Transform | FFT**. Execute **Format | Vertical Axis Scaling** and select **Linear** magnitude. **Zoom** around DC, and scroll the display to **M# 24** again.

Notice in Figure 8, that instead of removing the DC value, *it was made larger!*

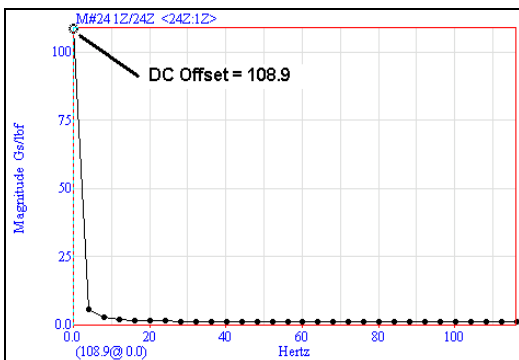


Figure 8. Offset of M#24 After Remove DC command.

Time Domain Integration

Now, lets try to integrate the new time domain waveforms with significant DC offsets in them.

- Execute **Transform | Inverse FFT** in the new Data Block to recover the IRFs.
- Execute **Tools | Math | Integrate** to integrate the IRFs.

A resultant velocity IRF is shown in Figure 9. What happened? When the signals were integrated, the DC offset *completely dominated* the IRF. This is the correct result, but not very usable. We would prefer velocities that still look like impulse responses.

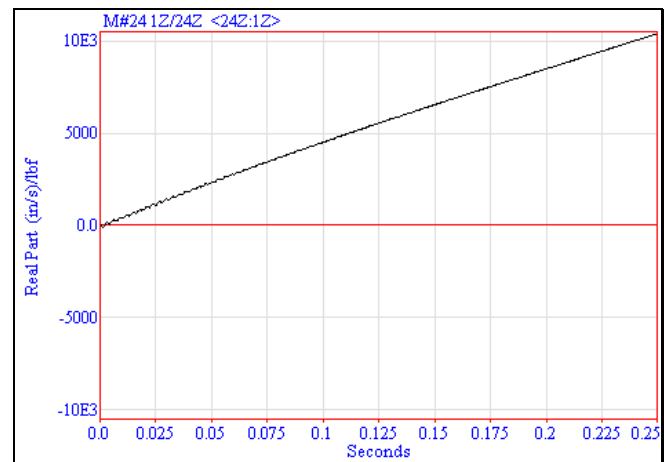



Figure 9. Velocity IRF Dominated by Large DC Offset.

Frequency Domain DC Removal

Now, let's clear out the DC offsets with the Band Pass window, and integrate the waveforms in the frequency domain.

- Close the **PLTMODES.BLK** window without saving it.
- Re-open the **PLTMODES.BLK** window by dragging the file from either the **top** or **middle** pane of the Project Panel onto the Work Area.
- Execute **Format | Vertical Axis Scaling** and select **Linear** magnitude. Then turn ON the Band Cursors .
- Drag the band cursors to enclose a band of approximately (180, 1050) Hz, as shown in Figure 10.

Now, when the band pass window is applied, all of the data outside of the cursor band will be zeroed, including DC.

- Execute **Transform | Window Data | Band Pass**. Click on **No** to window the data in the same Data Block.

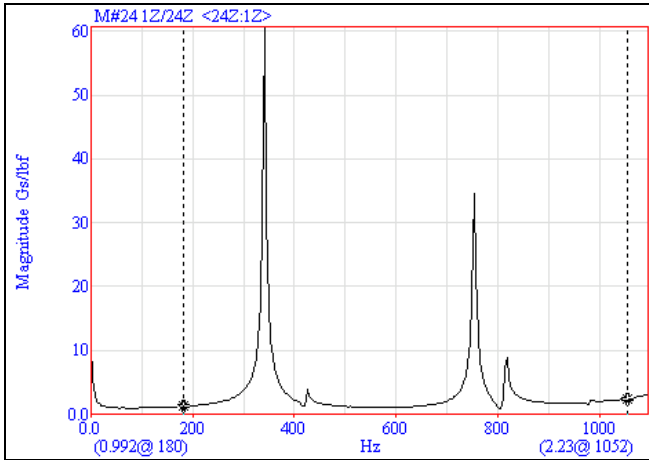


Figure 10. Band Cursors @ (180, 1052) Hz.

Frequency Domain Integration

To double integrate the signals in the frequency domain, and then recover the displacements in the time domain,

- Execute **Tools | Math | Integrate** command *twice*.
- Execute **Transform | Inverse FFT**.

The resulting Traces are now IRFs with displacement units. Notice that this time they still look like the expected impulse responses. M#24 is shown in Figure 11.

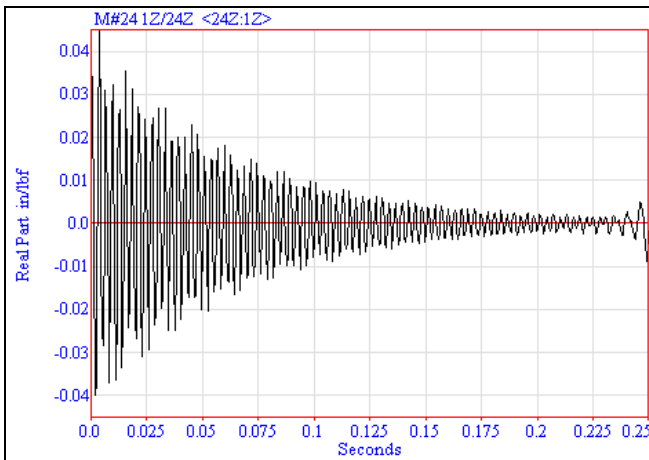


Figure 11. Displacement IRF.

WRAP AROUND ERROR

Even though the DC offsets have been effectively removed, and this result is a lot closer to the expected displacement result than the time domain result in Figure 9, it still exhibits a problem, called *time domain leakage* or *wrap around error*.

Notice that all of the IRFs exhibit the characteristically damped sinusoidal response, but then many of them begin to

grow in amplitude at the end of the window. Of course this is not realistic, since real vibrations don't damp out and then begin to grow again. Instead, this is a signal processing error.

This error is due to truncation of the signal in the frequency domain, the same as multiplying the true FRF by a rectangular window. (This effect is explained in more detail in Application Note #1.) The result is a smeared signal in the time domain. In this case, the IRFs were smeared by using the band pass window in the frequency domain to remove the low frequencies.

NON-PERIODIC TIME DOMAIN SIGNALS

From the last example, it might appear that using the FFT, clearing out low frequencies with band pass windowing, and frequency domain integration is the preferable method for integrating all signals. But, just as leakage caused a slight problem because of windowing the data (band passing it) in the frequency domain, it can cause an even greater problem when the original time domain signal is non-periodic in its sampling window.

NOTE: The IRFs created by using the Inverse FFT on **PLTMODES.BLK** are *nearly periodic in the window*, since they are *almost completely containing* within their time domain windows.

To create a sine wave that is non-periodic in its sampling window,

- Execute **File | New | Data Block** in the ME'scopeVES window. Name the Data Block file **3515.BLK** and click on **OK**. The **Synthesize Time Traces** dialog box will open.

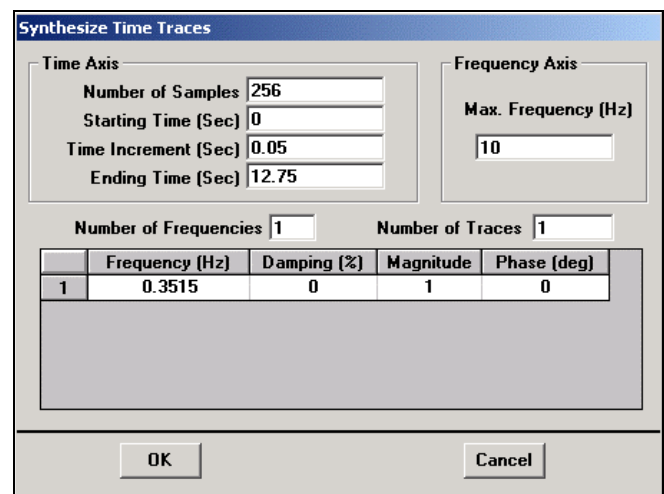


Figure 12. Dialog Box for Non Periodic Sine.

- Enter the following numbers into the dialog box (as shown in Figure 12),

Number of Samples: 256
Max. Frequency: 10 Hz
Number of Frequencies: 1
Number of Traces: 1
Frequency: 0.3515 Hz, Damping (%): 0
Magnitude: 1, Phase (deg): 0.

These parameters will synthesize a **0.3515 Hz** sine wave with a **magnitude** of **1.0**, and no damping.

- When all of the parameters are entered, press the **OK** button to synthesize the sine wave.

The Data Block window will open with the sine wave in it, as shown in Figure 13.

- Execute **File | Save** and click **OK** to save it to disk.

Time Domain Integration

To double integrate this waveform,

- Execute **Tools | Math | Integrate**.
- Execute **Tools | Math | Remove DC**, and select **No** to use the Min/Max method of removal.
- Execute **Tools | Math | Integrate** again.

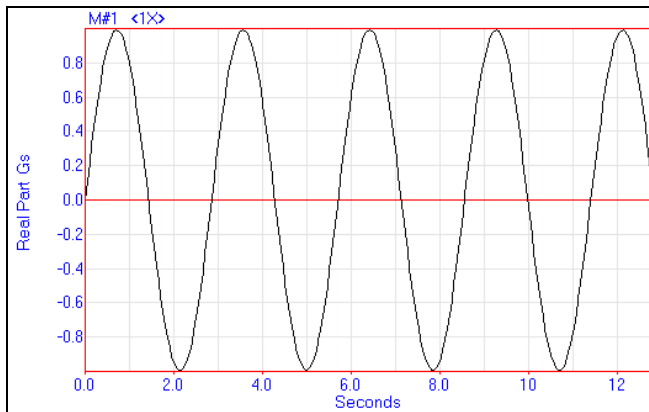


Figure 13. Sine Wave Non-Periodic in the Window.

A sinusoidal displacement waveform (the expected result) is shown in Figure 14. In this case, the Remove DC command worked correctly on a non-periodic signal, but only because DC was computed as the average of the minimum and maximum values. If all of the Trace data had been averaged together (by answering **Yes** during the **Tools | Math | Remove DC** command above), an incorrect DC value would have resulted.

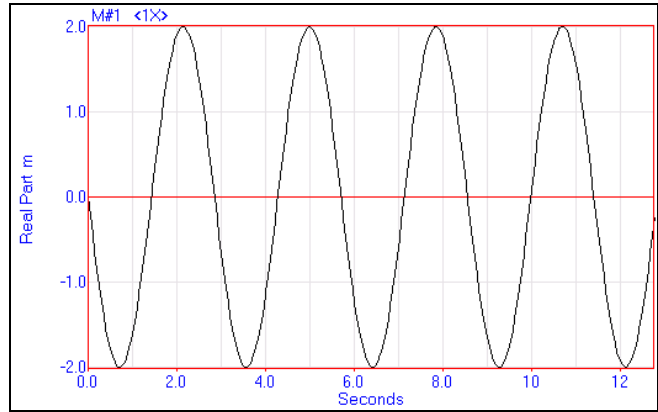


Figure 14. Double Integration of Non Periodic Sine Wave.

Frequency Domain Integration

Now, let's recall the non-periodic acceleration signal and use frequency domain integration to integrate it.

- Close the **3515** Data Block and don't save its changes.
- Execute **File | New | Data Block** again. Notice that all of the values you entered last time are still there. Click on **OK** to create the Data Block again.
- Execute **Transform | FFT** to obtain the Linear spectrum of the sine wave.
- Execute **Tools | Math | Integrate twice**.
- Execute **Transform | Inverse FFT** to obtain the displacement sine wave.

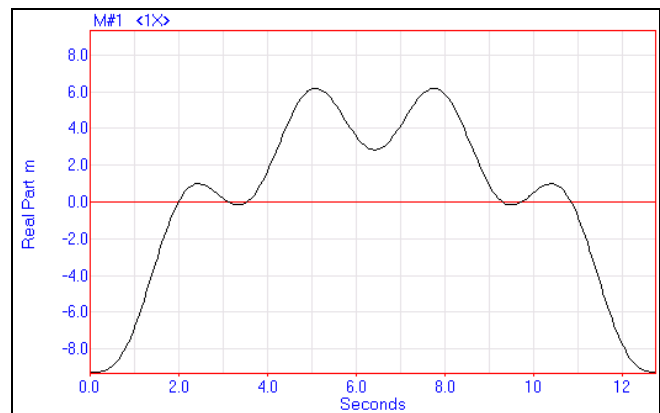


Figure 15. Frequency Domain Integration of a Non-Periodic Sine Wave.

The result is shown in Figure 15. What happened? Since the signal was non-periodic in the sampling window, severe leakage occurred, and the low frequencies in the spectrum, (distorted by leakage), were amplified by the double integration process.

Of course, a Hanning or Flat Top window could have been used to reduce leakage before the frequency domain integration. (See Application Note #1 for details.) The integrated frequency spectrum would be more accurate, but the resulting time waveform would show the effects of multiplication by the time window (truncation at both ends of the window).

To minimize the error due to double integration, a band pass window can be used before integrating the signal, to **remove as much of the low frequency leakage as possible**.

- Close the **3515** Data Block and don't save its changes.
- Execute **File | New | Data Block** again. Click on **OK** to create the Data Block again.
- Execute **Transform | FFT** to transform the sine wave data to the frequency domain.
- Execute **Format | Vertical Axis Scaling** and select **Linear** magnitude.
- Zoom in and set up the Band Cursors around the peak, as shown in Figure 16.

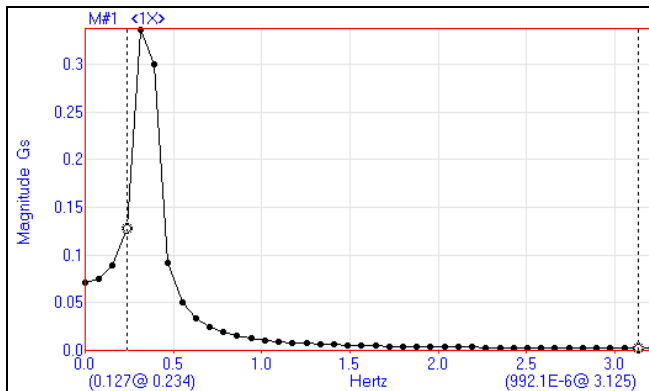


Figure 16. Band Cursors Surrounding Peak.

- Execute **Transform | Window Data | Band Pass** to preserve just the peak in the data. Select **No**, and click on **OK** to window the data.
- Execute **Tools | Math | Integrate twice**.
- Execute **Transform | Inverse FFT** to recover the integrated time domain signal, as shown in Figure 17.

This result is an improvement of that in Figure 15, but still not the expected result.

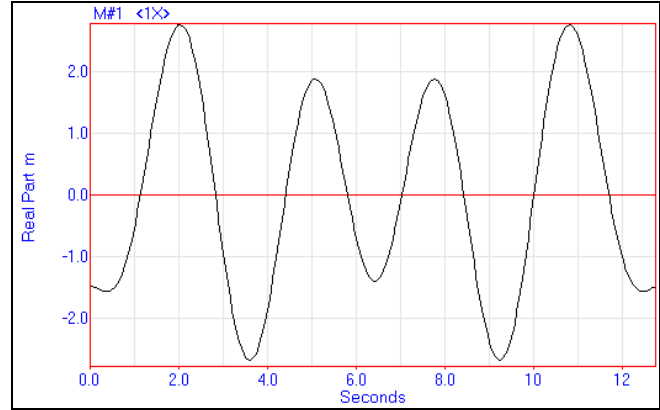


Figure 17. Displacement of Non Periodic Sine Wave After Band Pass Windowing of Frequency Data.

CONCLUSIONS

First, we saw that a sine wave that was **periodic in the sampling window**, could be integrated & differentiated using either time or frequency domain methods, and the same result was obtained.

Next we looked at the integration of impulse responses that were **nearly periodic** in the window, but had DC offsets in them. Since the IRFs were non cyclic, only band passing the data in the frequency domain could effectively remove the DC offsets.

Finally we saw the effects of leakage when trying to perform frequency domain integration on a **non-periodic** sine wave.

From these examples we can make the following conclusions,

- Time domain integration can be used on any time domain cyclic signal as long as DC removal is performed before integration.
- Frequency domain methods will also work well on time domain signals that are **periodic in the sampling window**.
- DC offsets of **non-cyclic** signals must be removed by band pass windowing the data in the frequency domain before integration can yield usable results.
- Before performing frequency domain integration of **non-periodic** waveforms, all non-essential low frequencies must be removed by band passing the data.