Aliasing and frequency resolution

Introduction

Part 2 said that design using impulse invariance gave results that were prone to problems of aliasing. So, 'aliasing' needs explanation. In engineering we have to make compromises and 'frequency resolution' is shown to call for similar criteria to those of aliasing, resulting in huge demands on resources. We generally have to compromise in some way.

Aliasing

DSP systems involve sampled data, so they are sampled data systems. The word 'sampling' implies regular discrete values being obtained from an analogue waveform. This usually means using an A to D converter. If we don't take a minimum number of samples of a given waveform, when we come to reconstruct that waveform results can be ambiguous as in fig 1.

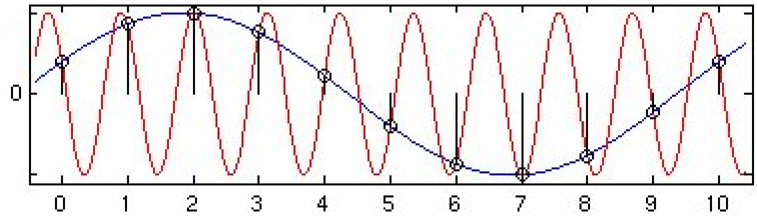

fig 1. Aliased sines (non copyright graph)

If the red waveform is sampled at too low a rate, as it is here, the sample points could reconstructed as in the blue waveform. If a filter is designed with a given sampling rate and, much higher frequencies appear at the input, then these spurious artefacts will get through the filter and cause problems because they have been interpreted as lower frequency components. Inputs need to be somehow 'band limited'. Another instance of the effect of too low a sampling rate is that short duration time domain signal features could 'fall between' samples and be missed. Remember that short duration features imply the presence of higher frequencies and to get the desired performance we really need to know the highest frequencies expected to be present in the wanted signal. To avoid the problems we need to take heed of Shannon's sampling theorem:

'The sampling frequency must be twice the highest frequency present in the signal.'

Half the sampling frequency, i.e. The highest frequency for which aliasing cannot occur is often called the **Nyquist frequency.**

Anti-aliasing filters are often included in systems to ensure that following circuits are never running at an 'under' sampling rate. Anti-aliasing measures would seem to advocate taking a lot of samples at a high rate. Hang onto this thought.

Frequency resolution

This is all about being able to follow slow changes with your chosen sampling rate. Think about the case of two sinusoidal waves quite close in frequency, both being present in the sampled signal. The longer you sample the easier and more likely it is that you will pick up the slow variation of the phasor sum of the two waves. Assuming you do pick up the slow variation and then translate the result into the frequency domain and the two components will be clear. If you don't sample for long enough the slow variation might be too slow to see and then in the frequency domain there will only be one component. An analogy is obtained by imagining a very low frequency sinusoid. If you sample it for just a short while you will see a straight line. No change=a single DC component. Sample it for a long time and the time domain samples will draw out a curve. My favourite illustration of 'frequency resolution comes from http://ccrma.stanford.edu/~jos/Intro421/Frequency_Resolution.html and is shown below as fig 2.

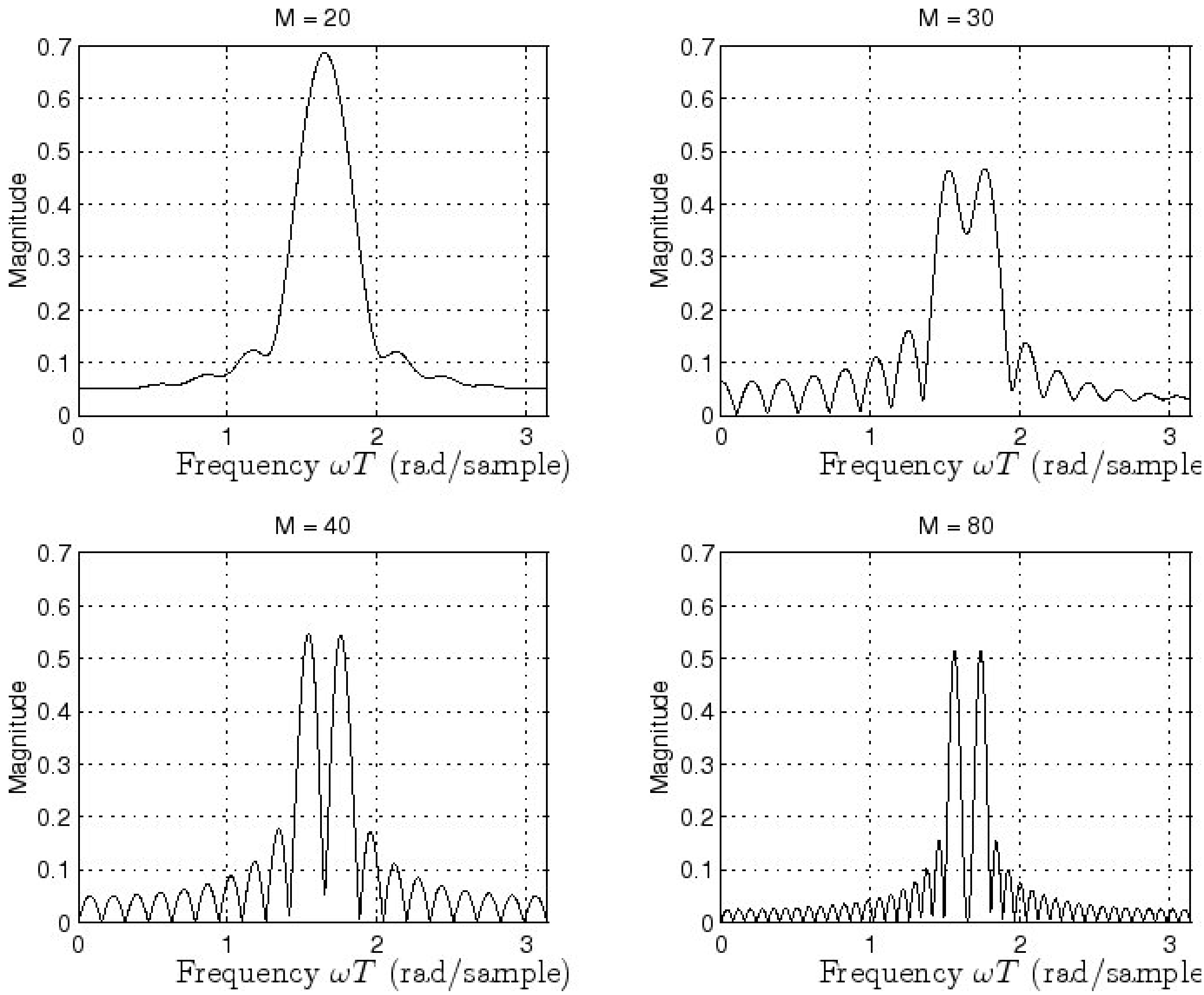

Fig 2

Fig.2 shows the frequency domain for two sinusoids 1/9 of a Herz different in frequency. In the diagram, M represents the sampling period. You can treat M as arbitrary units here but the important thing is that the sampling period for the last graph (bottom right) is 4 times longer than for the first (top left).

The essence of all this is that if you want to have a DSP system that recognises signal components that are close together you have to sample for a long time. i.e. Have lots of samples
Conclusion
So, anti-aliasing performance calls for lots of samples, i.e. A high sampling rate , and the ability to resolve the existence of frequency components that are close together calls for sampling for a long time. Both contribute to the need for dealing with 'lots of samples'. This, of course puts demands on the computer hardware. Lots of memory and very fast CPU's, together with high rate A/Ds and possibly D/As are the order of the day.

In the next part we shall look at Convolution and Correlation. Convolution and Correlation are similar, so what's the difference? Both can prove computationally expensive and time consuming, so what can be done to improve speed? Well, this calls for fast convolution and this will also be explained. However, fast convolution turns out to require FFT (Fast Fourier Transforms) and this in turn requires a CPU with a special Arithmetic Unit capable of performing FFT butterflies (as they are called). If you want high performance this also means parallel arithmetic units. This is the reason for the Texas DSP processor architectures, for SHARCs, and Blackfins! How does a PIC compare?