Data Acquisition, Processing and Analysis for Distributed Decision Support

Broad Contents

Chapter 2

Data Acquisition

2.1 Sources of Data

Data doesn't materialise from nowhere! As we saw in Chapter One, data is usually a physically measured value that is of interest to us. By measuring a value (data) from a system (source), we attempt to learn more about the system, i.e., the source from which data is collected. So, the primary objective of data acquisition and analysis is to know more about the data source and to characterise / classify / recognise / use the system effectively.

The data source could be of two types. In the first case, it could be a system that auto-generates the data, and in the second case, we interrogate the system with a probing medium, and collect the resultant data.

For the first case, here's a partial list of interesting systems (which generate data on their own):

- Rotating Machine Part – Rotational Frequency Data (Signal)
- Vibrating Machine – Machine Vibration Data (Signal)
- Human Brain – Electro-encephalograph Data (Signal)
- Market Price of a Stock – Stock Index Data (Signal)
- Pulsar (Spinning Neutron Star) – X-ray Data (Signal)
- Bats – Ultrasonic Data (Signal)
- Earthquakes – Seismic Data (Signal)
- Solar Black spots – Radio wave Emissions (Signal / Image)
- Arable Land – Agricultural Yield (Signal)
- Vegetation Cover – Remote Sensed Data (Image)
- Low and High Atmospheric Pressure Distributions – Weather Data (Image)
- Sonic Activity of Deforming Materials - Acoustic Emission (Signal)

For the second case, we generate a probing wave / particle and pass it through a medium and study the reflected / refracted / transmitted data, in which case the data source could be the medium or any artefact present in the medium. Examples for this second case are

- Ultrasonic testing of materials (ultrasonic waves as the probe; material tested as the system; reflected / refracted / transmitted ultrasonic waves as the collected data),
- Eddy current testing (electromagnetic waves as probe),
- Radiography (x-rays as the probe),
- Neutron radiography (neutrons as the probe) and so on.

Generally for both cases, and in particular for the second case, there are three phases before data is collected: (a) method used to generate (internally or externally) the probing wave / particle, (b) its transmission through and interaction with the probed material and (c) the way in which the reflected / refracted / transmitted probing wave / particle is collected. These three important aspects would determine the way in which the data should be interpreted*. While steps (a) and (c) are related to sensors and associated electronics, aspect (b) is largely a subject matter that falls under Physics. We presume that steps (a) through (c) are well understood and correctly implemented. In this Monograph we are interested in knowing what happens during and after the step (c), i.e., during and after data collection.

**See questions at the end of this Chapter*

The importance of *accurate* data collection, and the collection of *appropriate* data can never be over-emphasised. The success of any decision support endeavour depends crucially on accurate, appropriate data collection and in using the knowledge of the three phases (a) through (c) (mentioned in the last paragraph) in result interpretation. Hence, we shall now discuss more about types of data, and about data collection itself.

2.2 Analog and Digital Data

Let us examine Figure 2.1 given below:

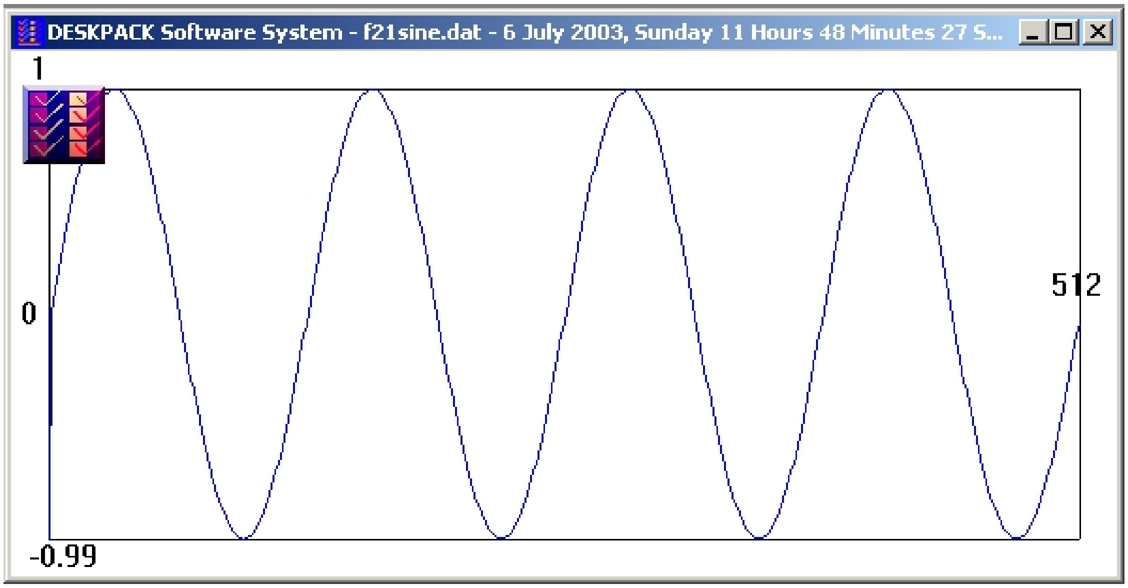
DESKPACK Software System - f21sine.dat - 6 July 2003, Sunday 11 Hours 48 Minutes 27 S...
1
0
512
-0.99

Figure 2.1 A Typical Sine Wave

As seen in Chapter 1, Figure 2.1 shows a sinusoidal wave, in which the X-axis can be thought of as representing time T, while the Y-axis can be thought of as representing a measured value, say Magnitude M. So, the figure shows Magnitude as a function of Time – it represents how the value of M (Magnitude) changes, as time T increases. In the Physical World where such measurements are made, both M and T vary *continuously* – i.e., for any two given magnitude values m_1 and m_2, measured at t_1 and t_2 respectively, we can always find a value m_3 measured at t_3, such that $t_1 < t_3 < t_2$. This implies that *between* any two time intervals t_1 and t_2, however close they are, we can always make a meaningful measurement at t_3, leading to the measured values, m_1, m_2 and m_3. You must note that the measured value m_3 at the time instant t_3, may or may not satisfy the inequality $m_1 < m_3 < m_2$.

The direct implication of this statement is that between any two instants of time, say t_1 and t_2, there could be *infinite* number of time instants at which the measurements on Magnitude could be made! Such continuously varying data are fully represented in the natural world as analog data – where both the measured value M and the Time T are analog data.

If so, how to represent an infinite number of data points (infinite number of T values and the corresponding infinite number of measured M values) in a *finite* resource such as a Computer? How to store, process and understand an infinite amount of data? Is it necessary to represent the infinite data as it is, in order to understand it? If we decide to represent only a finite part of this infinite data, would there be any information loss? How to arrive at that finite portion from this infinite data?

Fortunately, it is possible to *sample* only a finite amount of information from an infinite train of analog data and use a Computer to process and understand this sampled data, without much information loss. The trick is to take samples of the measured value M, at certain intervals of time T. This process is known as digitisation. In fact, any Computer-reproduced information, such as the Figure 2.1, shows only a finite amount of sampled information, which is nothing but digital data.

However, this process of digitisation for a signal like the one shown in Figure 2.1 is usually done at two levels, one for the X-axis and the other for the Y-axis. It is *not* necessary to digitise both the X- and Y-axis variables, though the normal practice is to digitise both the dependent (Y-axis) and the independent (X-axis) variables. If only the X-axis (say, time) is digitised (i.e., the value of M is measured only at certain time intervals), but the Y-axis value of M is kept as it is (as an analog floating-point value), the resultant data is known as *discrete* data. On the other hand, if both the X-axis and the Y-axis data are digitised, then the resultant data is known as *digital* data.

Figure 2.2 shows a typical digital data, where a portion (a crest of the sine wave) of Figure 2.1 is zoomed to show the effect of sampling and digitisation.

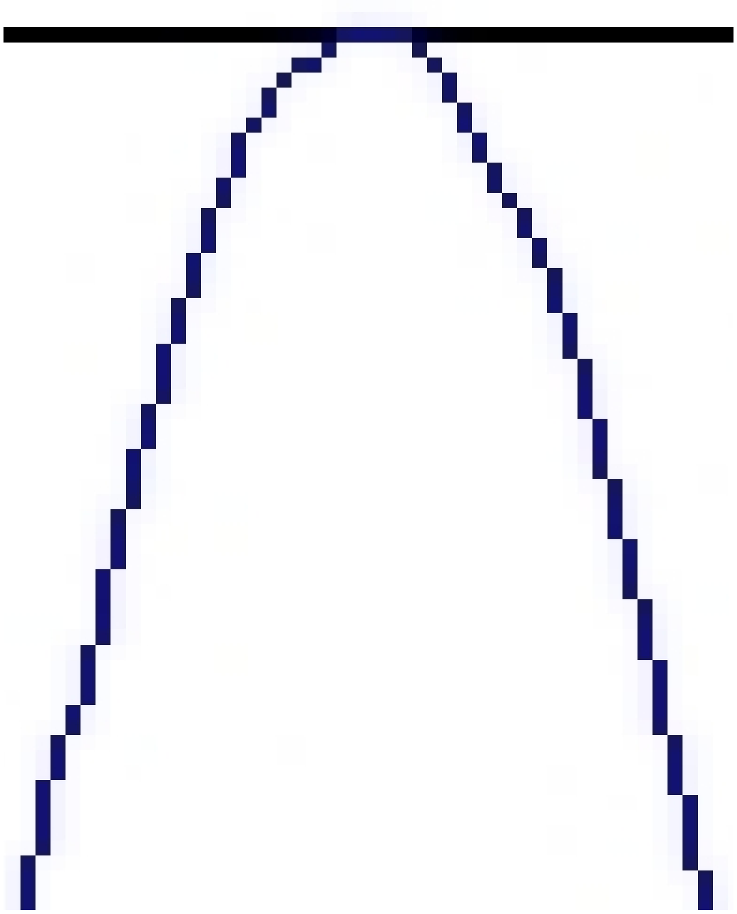

Figure 2.2 Zoomed portion of Figure 2.1, which shows the effect of sampling or digitisation

As you can see from Figure 2.2, measurements of M are made at *discrete* time intervals, and that the sampled values of M are simply connected by straight lines. This *process* of digitising analog data to obtain digital data is perfectly acceptable under certain *conditions*.

It is interesting to note that these principles of digitisation hold good remarkably for images too. In the case of images, a single "point" which represents an image is known as a picture element or a pixel. We then talk about the resolution of an image as having X-number of pixels horizontally and Y-number of pixels vertically. Let us learn more about the digitising *process* and the *conditions* in the next few Sections.

2.3 Hardware for Data Acquisition and Digitisation

Data *acquisition* and *digitisation* is always handled by dedicated hardware, specially designed for this purpose. Data *processing*, however, could be done either by hardware or through software. For continuous incoming data, this involves two processes, viz., sampling the data value and digitising it, before the next data point arrives. Clearly, higher the data throughput, higher should be the required processing power. Hence, specialised hardware becomes necessary for digitisation.

Hardware for data acquisition and digitisation could either be a Card, which will fit into a Personal Computer as a daughter board, or it could be a standalone, independent system.

Depending upon whether a signal (one dimensional data as in Figure 2.1) or an image (two dimensional data as in Figure 1.6) is acquired, the hardware's required processing capability is estimated.

More information on hardware-related aspects of data acquisition and digitisation can be read at [2].

2.4 Sampling and Digitisation

When we measure the Y-axis value at discrete intervals of time (at t_1, t_2, t_3,…t_n), we say that we *sample* the Y-axis value (to get $y(t_1)$, $y(t_2)$, $y(t_3)$,…$y(t_n)$). The number of *samples* we measure per second is then called the *sampling frequency*, S_f. The time intervals (t_2-t_1), (t_3-t_2),…(t_n – t_{n-1}) are normally identical, and are called the *sampling period*, S_t. The sampling period and the sampling frequency are *inversely* related to each other, i.e.,

$$S_t = 1 / S_f \qquad \text{(Equation 2.1)}$$

The sampled value y(t) – value of 'y' at time 't' – is still an analog value (floating-point value), till we digitise it – to be represented as a sequence of '0's and '1's. The number of zeroes and ones we use to represent an analog value is called the number of 'bits' used for digitisation.

For example, a single bit is either a single 0 or 1. With a single bit (N=1) we are able to represent only two values (V=2). If we use two bits, we can write four sequences such as 00, 01, 10 and 11. So, with two bits (N=2) we are able to represent four values (V=4). If we use three bits, we can write eight sequences such as 000, 001, 010, 011, 100, 101, 110 and 111. So, with three bits (N=3) we are able to represent eight values (V=8).

Note that V = (2 raised to the power N), i.e., $V = 2^N$

In a typical case, assume that we need to measure a voltage value that varies anywhere from 0.0 volts to 8.0 volts. So, y(t) could be 0.0 volts, 0.4 volts, 5.4 volts, 8.0 volts, 6.2 volts and so on. The minimum value which y(t) can take is 0.0 and the maximum value it can take is 8.0 volts. Hence, we say that the *range* of y(t) is maximum value – minimum value, i.e., 8.0 – 0.0 = 8.0 volts. If we use a 3-bit digitiser discussed above, we can represent eight values, and so we can represent y(t) = 0.0 as 000 and y(t) = 8.0 as 111. All other values of y(t) [0.0 < y(t) < 8.0] are represented by one of those eight bit-sequences viz., 000, 001, 010, 011, 100, 101, 110 and 111. Since we have only eight distinct values to represent, a discerning reader would immediately find that a measured voltage value of say 0.6 to 1.5 would all be represented by the sequence 001! In other words, for a range of 8.0 volts, if we use a 3-bit digitiser, our error in representation could be 8.0 / 2^3 volts or 1.0 volt. For precise representation of measured values, this is clearly unacceptable. One way to overcome this deficiency is to increase the number of bits used to represent measured values.

It is a normal practice to use either 8-bit, or 12-bit digitiser. Measurements that require higher precision use a 16-bit or a 32-bit digitiser. For example, if we use a 8-bit digitiser, we will have 2^8, or 256 distinct values to represent a range of measured values. So, in the previous example, if we use a 8-bit digitiser to represent a range of 8.0 volts, our error in representation would only be 8.0 / 2^8 volts or 0.03125 volts. So, as we increase N (the number of bits used), we will be able to represent measured values with better and better accuracy. If we use N-bits to represent a value, we are said to use an N-bit digitiser.

However, you must remember that as we increase the number of bits, we also need better processing power to sample and digitise the measured values, store the digitised values and then get on to sample the next measured value quickly.

How *quickly* should we get on to measure the next value, during digitisation?

That is, if we measure the values $y(t_1)$, $y(t_2)$, $y(t_3)$, $y(t_4)$ and so on, what should be the time interval – the sampling period - between successive measurements? That is, what should be the value of t_2-t_1, t_3-t_2, or t_4-t_3? We saw that the sampling period is inversely proportional to the sampling frequency. The acceptable sampling period - and hence the sampling frequency – of a digitisation process is governed by the Nyquist criteria, which says,

If in a measured process, the largest frequency of occurrence is ***f*** then the sampling frequency should *at least* be ***2f***.

What this criteria means is that if we have a sinusoidal wave whose maximum frequency content **f** is 50 cycles per second (50 Hertz), we must sample the wave at **2f**, i.e., at 100 samples per second (100 Hz), to avoid information loss. A good example of such a wave is the normal domestic power supply frequency in many parts of Asia and Europe. If we sample this wave at a sampling frequency less than 100 Hz, a phenomenon called *aliasing* occurs that leads to information loss.

However, in real-world situations the sampling frequency S_f is usually ten times or more than the highest frequency content in the sampled wave. Higher the sampling frequency, higher would be our ability to digitally represent the measured value accurately.

Figures 2.3 and 2.4 show the effect of higher sampling rate. In Figure 2.3, the basic sine wave (which we wish to sample) is shown in blue. Note that there are four crests (peaks) and four troughs (valleys). Roughly, since there are four peaks, let us just use the Nyquist criteria and sample this wave at 2f, i.e., at just eight points. These eight points where we sample the wave are shown as red points. If we join the red points to obtain the waveform (black line), we a get a very rough triangular wave, that is not much similar to the original sine wave. Still, this rough triangular wave gives an indication about the oscillation pattern of the original sine wave.

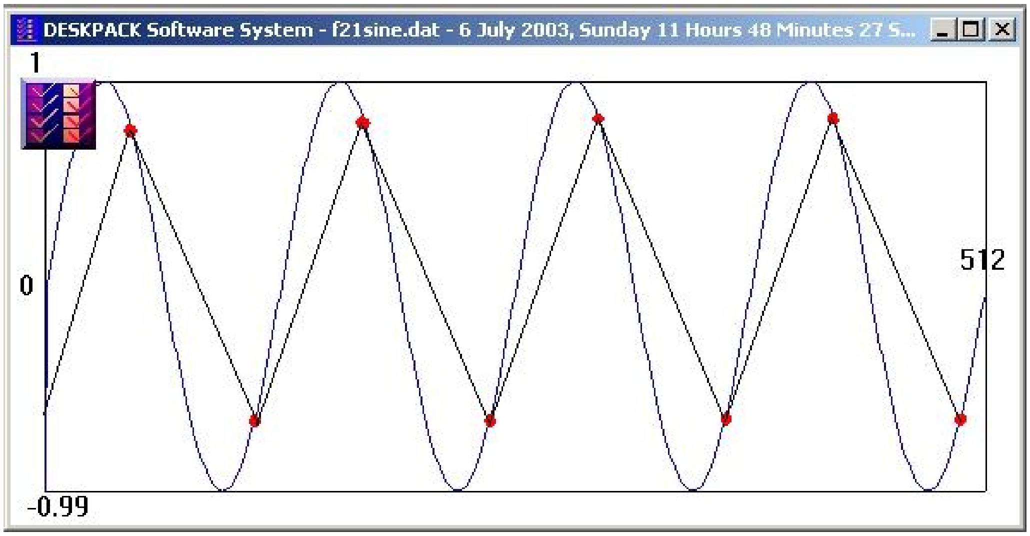

Figure 2.3 Sine wave sampled at just 2f to satisfy the Nyquist criteria (Low sampling)

What happens if we increase our sampling rate to about ten times the highest frequency? That is, for each crest (or trough) we sample it ten times. Figure 2.4 shows the effect. If we join the red dots (sampling points) in Figure 2.4, they would almost merge with the blue line, offering a better representation of the sine wave that is being sampled. This is shown in Figure 2.5.

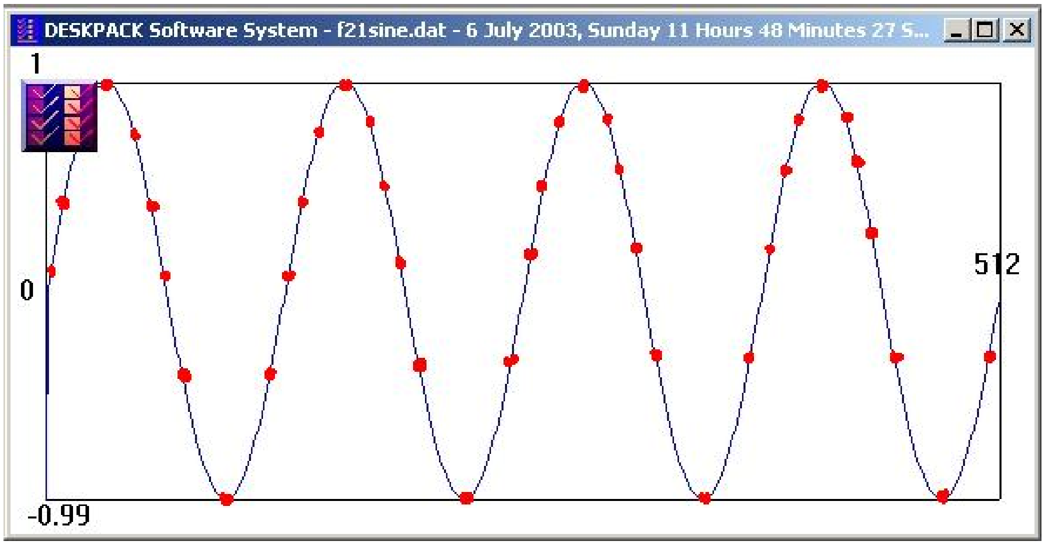

Figure 2.4 Sine wave sampled at just 10f (High Sampling Rate)

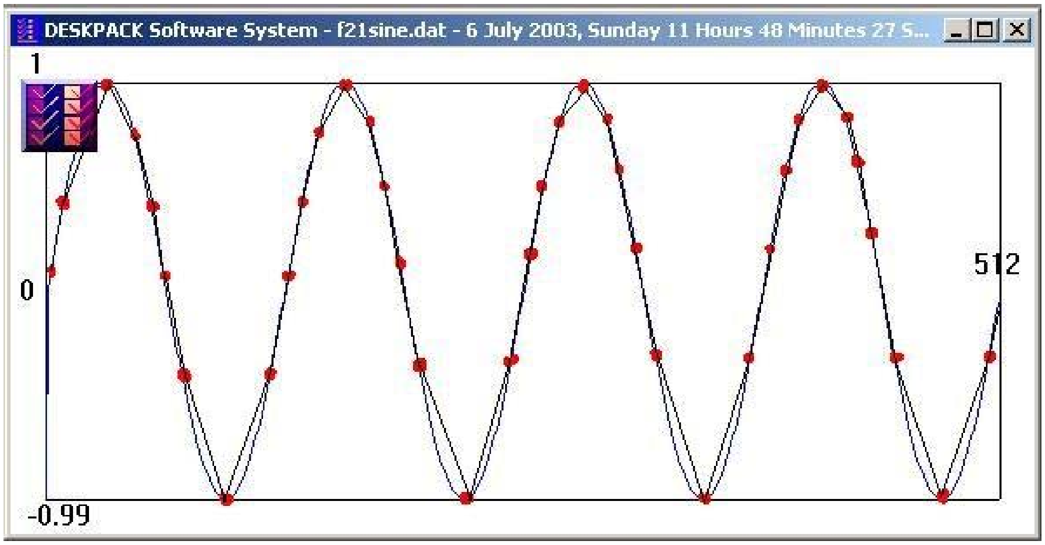
DESKPACK Software System - f21sine.dat - 6 July 2003, Sunday 11 Hours 48 Minutes 27 S...
1
0
512
-0.99

Figure 2.5 Connected waveform (Black Line) of High Sampled Waveform

2.5 *Representation of Digital Data -- Signals, Images, Databases, Spreadsheets*

Once data is digitised, it needs to be stored for future representation and use. Representation of digital data is a very important aspect of data processing and analysis, and can broadly be studied from at least three perspectives, viz., (a) the internal representation of data inside a computer, (b) the external representation of data for human consumption (in the form of an image, waveform, database, spreadsheet, etc.) and (c) the interfacing of digitised data with computer systems and programs for either further processing and decision support.

The internal representation of digital data inside a computer can either be in ASCII (refer Figure 1.8) or as a binary file. Binary files occupy less space as compared to ASCII files, but are not human-readable. Digital data can also compressed and encrypted so as to occupy lesser space and for better security, respectively.

The external representation of digital data will always be in a form that aids easy understanding by a human being. This could be a signal or an image, as we have seen in the previous Chapter. Raw data could also be represented in the form of spreadsheets and/or databases to get a feel of the depth and extent of the data. Representation of digitised data in the form of spreadsheets and databases also gives us a better handle to process them programmatically. Chunks of digital data that represent a single entity (a single signal or a single image) are normally stored in a single file with appropriate file extension. Data files normally have a *.dat file extension, while images could have a number of file extension possibilities, the most effective in terms of storage being the JPEG (*.jpg) file.

In situations where digitised data need to be interfaced with other a number of computer programs and/or systems that access the data sequentially for processing and analysis, the data must be represented in a mutually acceptable format, e.g., ASCII. However, if disparate number of systems accesses the data, there needs to be an established, standard way to represent data for processing and analysis. Standard representation of data becomes even more important, if independent computer systems arrive at decisions based on the analysis-outcomes of such data. One emerging standard method to represent data for heterogeneous computer systems is the eXtensible Markup Language or XML.

We shall study more about data representation and its relation to decision support in Chapter Five.

2.6 *Questions*

- Identify other sources of data in nature
- If we represent the (a) temperature measured at a city over a period of time, and the (b) value of a stock over the same period of time, what would be the difference in the domain expertise needed to interpret these data?
- If *accurate* data collection relates to the hardware used, what does *appropriate* data collection relate to?
- For any two given magnitude values m_1 and m_2, measured at t_1 and t_2 respectively, we can always find a value m_3 measured at t_3, such that $t_1 < t_3 < t_2$. But why is it that the inequality $m_1 < m_3 < m_2$ may not hold good?
- Name an equipment that processes the received data completely in analog mode.
- If increasing N, the number of bits used in digitisation, improves accuracy, can we increase N to a very high value (nearly infinity) for the best accuracy possible? What are the consequences of this increase?
- Should the sampling period (value of t_2-t_1, t_3-t_2, or t_4-t_3) be constant? What if not?
- What is *sampling jitter* and *aperture error*?
- Can you think of a physical example of aliasing?
- Assuming that we have identified the appropriate sampling rate to digitise an analog data stream and have a digitiser to perform the job. For how long should we keep digitising? Assuming that we decide to digitise in "chunks" (each "chunk" having certain *record length*), will this "chunk" depend upon the *type* of data stream we are digitising?
- A pure sinusoidal wave or white noise carries no information. Comment.