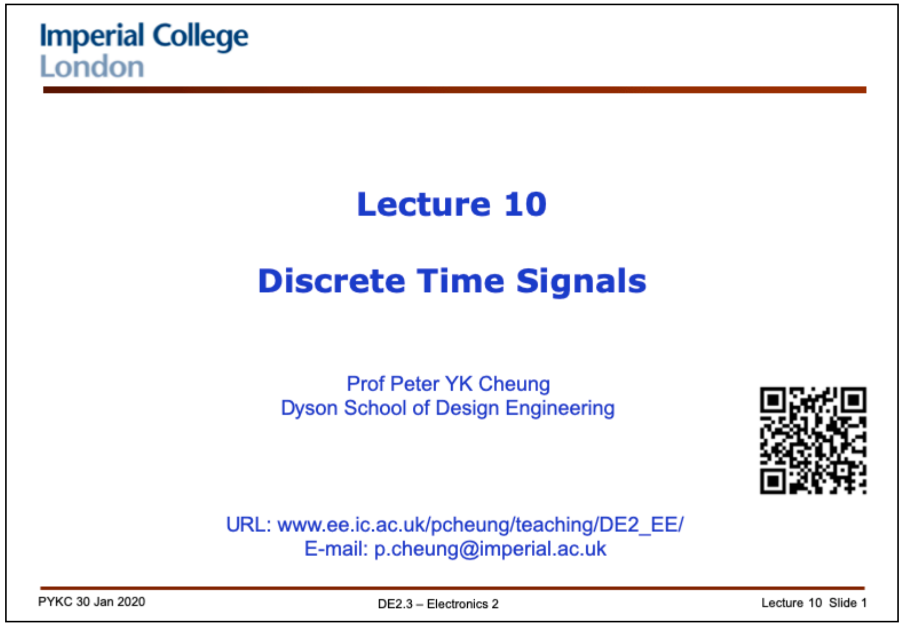


Imperial College London

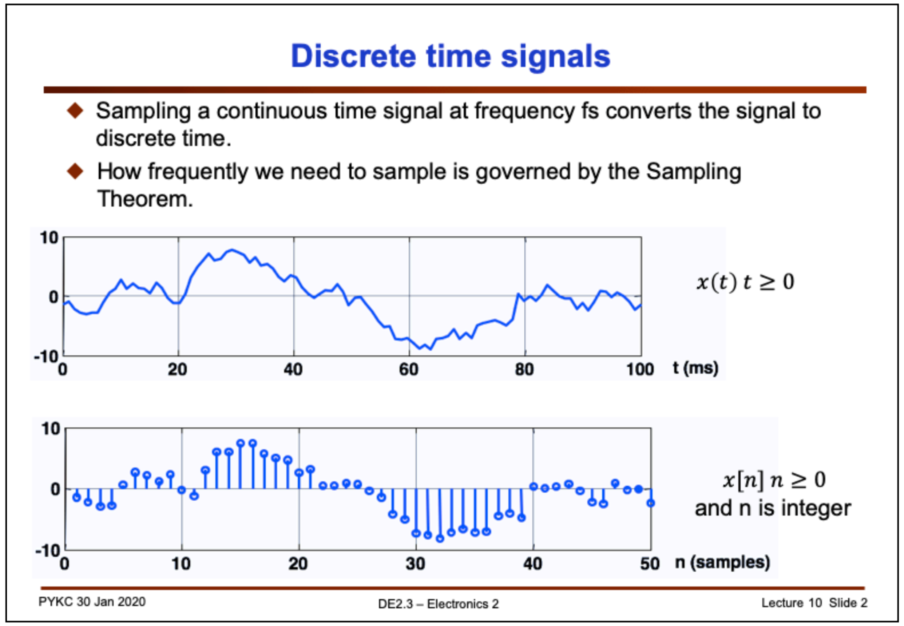
Lecture 10

Discrete Time Signals

Prof Peter YK Cheung
Dyson School of Design Engineering

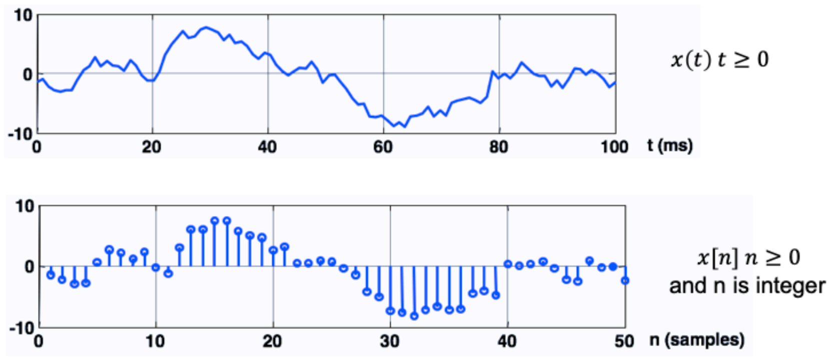
URL: www.ee.ic.ac.uk/pcheung/teaching/DE2_EE/
E-mail: p.cheung@imperial.ac.uk

In this lecture, I will introduce the mathematical model for discrete time signals as sequence of samples. You will also take a first look at a useful alternative representation of discrete signals known as the z-transform.



Discrete time signals

- Sampling a continuous time signal at frequency fs converts the signal to discrete time.
- How frequently we need to sample is governed by the Sampling Theorem.

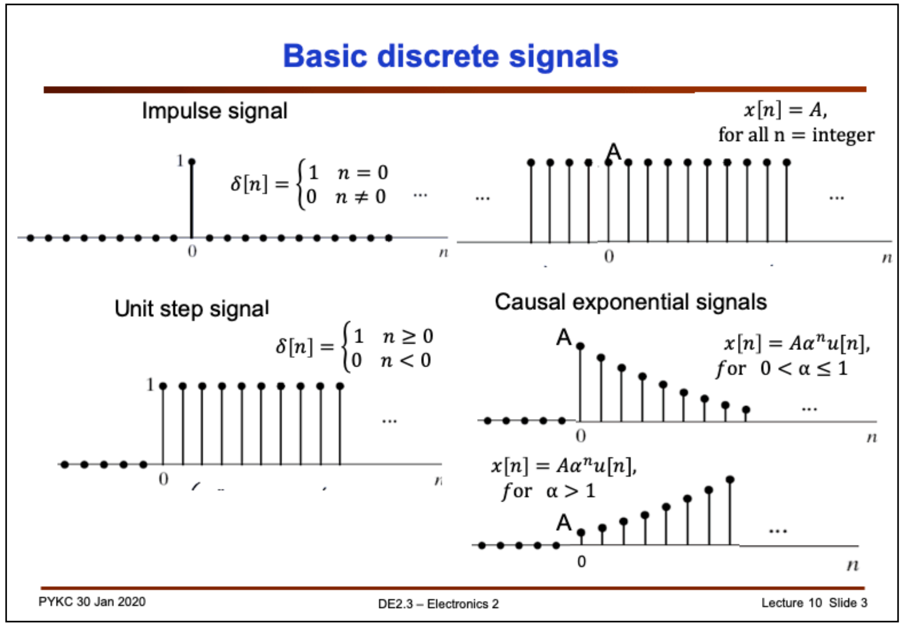


You are already familiar with the idea of sampling (using a ADC) to convert a continuous time signal to discrete time.

The important piece of information you need to known in this conversion process are:

1. **How to chose the sampling frequency fs**? The answer to this question is that, based on the Sampling Theorem, you need to use fs ≥ 2 fmax, where fmax is the maximum frequency component of the signal. If you do not obey the Sampling theorem, frequency components higher than fmax will be folded back to the lower frequency range – a phenomenon known as 'ALIASING'. In practice, we generally use fs ≥ 4 fmax or higher. Therefore when handling discrete signals, you must remember the sampling frequency fs and therefore the sampling period Ts. Everything you do to the signal will depend on this.
2. **How many bits to use to represent each data sample?** This is the number of bits that the ADC provides, i.e. its bit resolution. The answer to this question depends on the amount of quantization noise you are willing to tolerate. For example, if you are dealing with normal speech signal, 10-bit resoluation would generally be good enough. However, if you are in a recording studio, trying to capture a chamber orchestra performing a piece of classical music by Mozart, you may need 20-bit resoluation or higher in order to have a very high quality recording of the performance. If an ADC has N bit resolution, then the signal-to-noise ratio of the digitised signal (i.e. signal/noise) would be around 6xN dB:

$$20 \log_{10} \text{(signal voltage)/(noise voltage)} = 6N \text{ dB}$$



Basic discrete signals

Impulse signal

$$\delta[n] = \begin{cases} 1 & n = 0 \\ 0 & n \neq 0 \end{cases}$$

$x[n] = A,$ for all n = integer

Unit step signal

$$\delta[n] = \begin{cases} 1 & n \geq 0 \\ 0 & n < 0 \end{cases}$$

Causal exponential signals

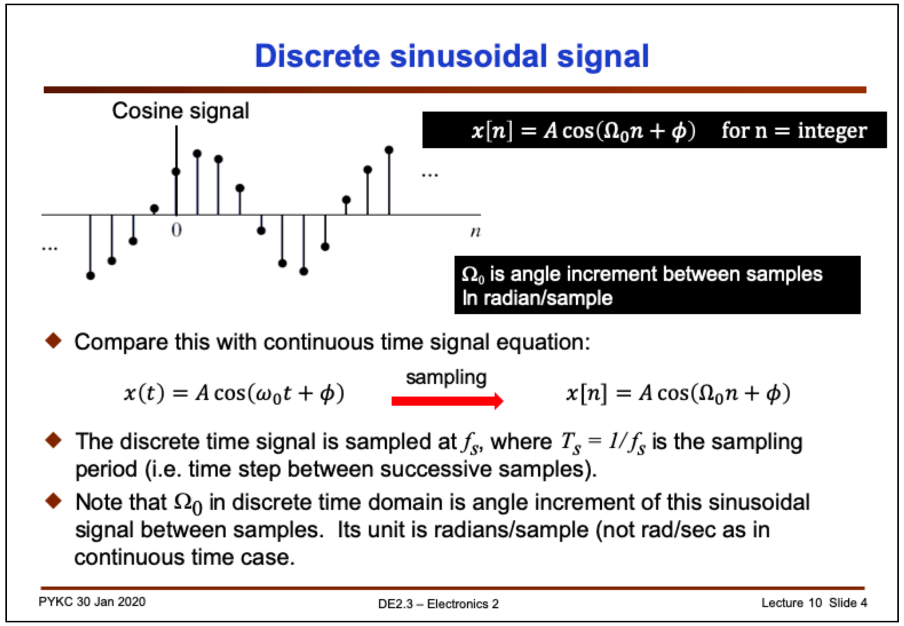
$x[n] = A\alpha^n u[n],$ $for\ \ 0 < \alpha \leq 1$

$x[n] = A\alpha^n u[n],$ $for\ \ \alpha > 1$

Here are FOUR basic signals and their discrete representation.

1. **Unit impulse** – This is represented by the discrete delta function. When n = 0, δ(0) = 1, otherwise δ(n) = 0.
2. **DC voltage** – This is straight forward. However, note that even the DC voltage is sampled, and the signal is represented as x[n] = A, where A is the voltage value. This is therefore a sequence of samples: {A, A, A, A, …}.
3. **Unit step signal** – This is represented by the sum of lots of unit impulses, at each sampling points for n ≥ 0. For n < 0, x[n] = 0.
4. **Exponential signal** – This is another important signal. Here we assume that the signal is causal (meaning is 0 before the t origin). Note that whether the signal is exponential rise or fall depends on the value of the constant α. If 0 < α ≤ 1, x[n] is an exponent decaying signal. If α > 1, x[n] is an exponent growing signal.

Note that while we use the notion with parentheses x(t) to represent a signal in continuous time, we use square brackets x[n] to indicate that the signal is now represented as a sequence of numbers. n is the sequence count or sample number (from 0 up), and x[n] is the value of that sample n.



Discrete sinusoidal signal

Cosine signal

$x[n] = A\cos(\Omega_0 n + \phi)$ for n = integer

Ω_0 is angle increment between samples In radian/sample

- Compare this with continuous time signal equation:

$$x(t) = A\cos(\omega_0 t + \phi) \xrightarrow{\text{sampling}} x[n] = A\cos(\Omega_0 n + \phi)$$

- The discrete time signal is sampled at f_s, where $T_s = 1/f_s$ is the sampling period (i.e. time step between successive samples).
- Note that Ω_0 in discrete time domain is angle increment of this sinusoidal signal between samples. Its unit is radians/sample (not rad/sec as in continuous time case.

Another important discrete signal is the sinusoid. Consider the following cosine signal with amplitude A, at frequency ω_0 and phase Φ.

$$x(t) = A\cos(\omega_0 t + \phi)$$

The discrete signal can be mathematically modelled as:

$$x[n] = A\cos(\Omega_0 n + \phi) \quad \text{for n} = \text{integer}$$

Note the following important differences:

1. In continuous time, t is a continuous quantity. Therefore there are infinite values for x(t). In contrast, x[n] is only definited for integer values of n.
2. In continuous time, we use the angular frequency ω_0 which has a unit of rad/sec. In discrete time we use the angular phase increment between samples Ω_0, which has a unit of rad/sample. We call Ω_0 the discrete frequency of the signal.
3. The relationship between ω_0 and Ω_0 is:

   $\Omega_0 = \omega_0$ x Ts, where Ts is the sampling period = 1/fs.

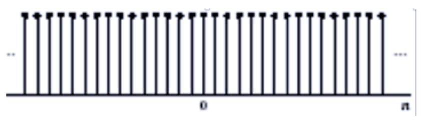
This concept of signal frequency in discrete time is something that many students found difficult. It is not helped by the fact that many textbook erroneously use ω_0 for both continuous time and discrete time.

Frequency is the measure of change in signal phase ϕ over time t, i.e. f = dϕ/dt. It unit is therefore rad/sec. Since we have discretized time into steps of Ts, we should now measure change in signal phase per sample. Hence the unit of Ω_0 is rad/sample, not rad/sec.

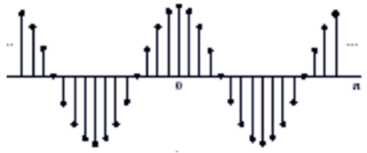
Discrete sinusoidal signal

$x[n] = A\cos(\Omega_0 n + \phi)$ for n = integer

Ω_0 is in rad/sample

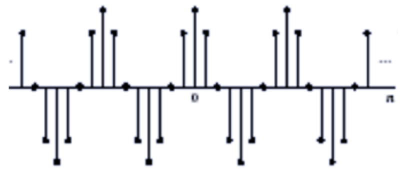


(a) When $\Omega_0 = 0$ *or* $x[n]$ is DC



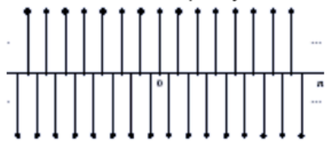
(b) When $\Omega_0 = \frac{\pi}{8}$

i.e. 16 sample/cycle



(c) When $\Omega_0 = \frac{\pi}{4}$

i.e. 8 sample/cycle



(d) When $\Omega_0 = \pi$

i.e. 2 sample/cycle

Let us explore the significance of Ω_0 a bit more.

Signal (a) – This represents a DC voltage. This is effectively having $\Omega_0 = 0$.

Signal (b) - Now let us consider the case where $\Omega_0 = \pi/8$. Interestingly, we don't really worry about Ts here. Instead, we consider how much the signal phase angle is advance every successive sample. Since a sinusoidal signal has a phase 2π for each cycle, having $\Omega_0 = \pi/8$ implies that we take N samples per cycle, where N = 16.

What is the frequency $f_0 = \omega_0/2\pi$ of the original signal x(t) (before we convert it to discrete time)? We can only answer this if we know the sampling frequency fs.

Suppose the sampling frequency fs = 8kHz, (i.e. Ts = 0.125msec) and there are 16 samples per cycle. Therefor the signal fo = fs/N = 500Hz.

(Make sure you understand how to relate sampling frequency to signal frequency and number of samples per cycle in the context of discrete signals.)

Signal (c) – In this case, $\Omega_0 = \pi/4$. We are therefore taking 8 samples per signal cycle, or N = fs / fo = 8.

Signal (d) - $\Omega_0 = \pi$, and each cycle of the signal has an angle of 2π. We are therefore sampling at twice the signal frequency – i.e. we are at the limit of the allowed minimum sampling frequency. This frequency beyond which signal cannot be captured without aliasing or frequency folding $f_q = f_s/2$, is known as the **Nyquist frequency**.

One important note: these diagrams show special cases where fs/fo is a whole integer. Then the discrete time signal is a periodic signal. In general, a discrete time version of a sinusoid or any periodic signal is NOT periodic, unless fs/fo is an integer and we have integer number of sample for signal cycle.

Operations on discrete signals

- Sum of two signals:

$$s[n] = x[n] + y[n]$$

- Product of two signals:

$$p[n] = x[n]\,.\,y[n]$$

- Amplification of a signal:

$$y[n] = \alpha\,.\,x[n]$$

- Delaying a signal by k samples:

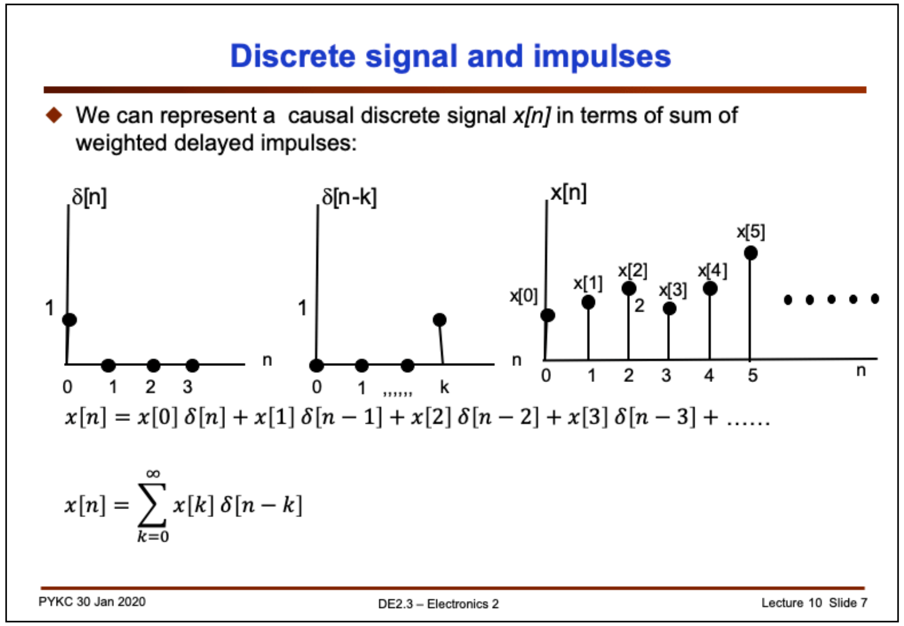
$$y[n] = x[n-k]$$

Here we have the four most basic operations applied to discrete signals as sequence of sample values.

The most important operation is the delay operator.

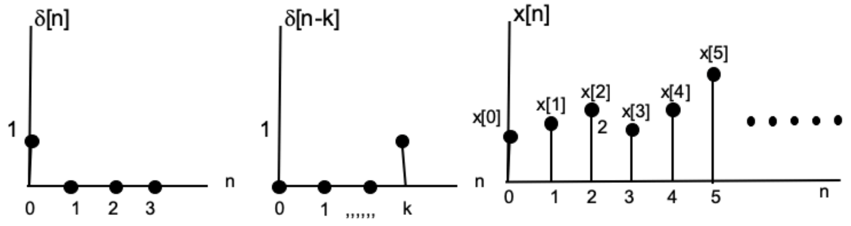
$$y[n] = x[n-k]$$

Here y[n] is the x[n] sequence delayed by k sampling periods.



Discrete signal and impulses

- We can represent a causal discrete signal *x[n]* in terms of sum of weighted delayed impulses:



$$x[n] = x[0]\,\delta[n] + x[1]\,\delta[n-1] + x[2]\,\delta[n-2] + x[3]\,\delta[n-3] + \ldots\ldots$$

$$x[n] = \sum_{k=0}^{\infty} x[k]\,\delta[n-k]$$

Mathematically, it is really useful to model a discrete time signal x[n] consisting of sample sequence: {x[0], x[1], x[2] …} in terms of the unit impulse with different delays.

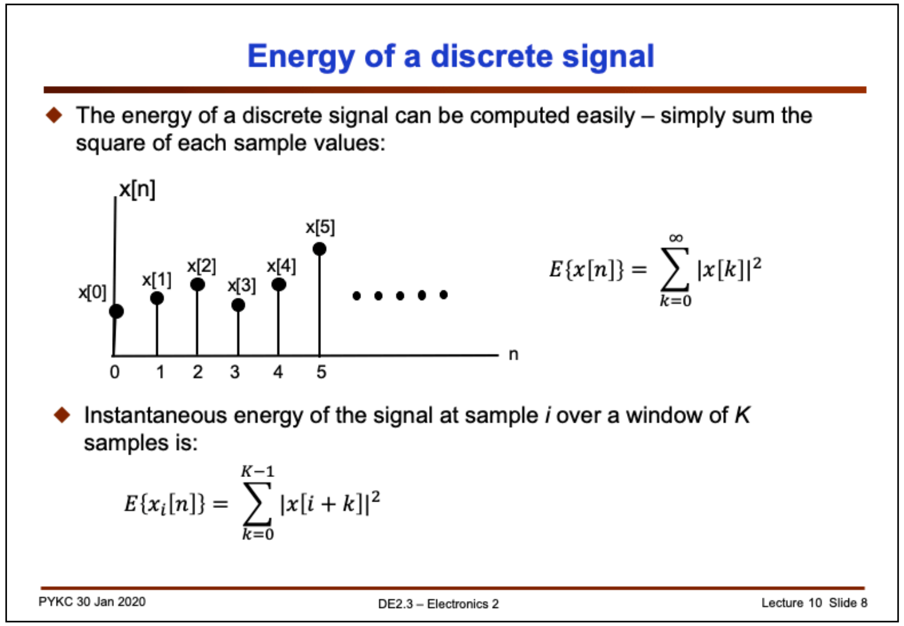
The left hand plot is the unit impulse δ[n]. The plot to its right (middle) is the unit impulse delayed by k sample intervals.

Equipped with this, we can decompose the sequence x[n] into a sum of delay impulses at each sampling point, each being weighted by the signal x[n].

How does not relate x[n] to the original continuous time signal x(t)?

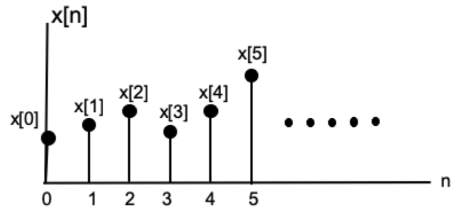
x[n] = x(nTs), where Ts is the sampling period = 1/fs.

Note that we assume that signal is causal.



Energy of a discrete signal

- The energy of a discrete signal can be computed easily – simply sum the square of each sample values:



$$E\{x[n]\} = \sum_{k=0}^{\infty} |x[k]|^2$$

- Instantaneous energy of the signal at sample *i* over a window of *K* samples is:

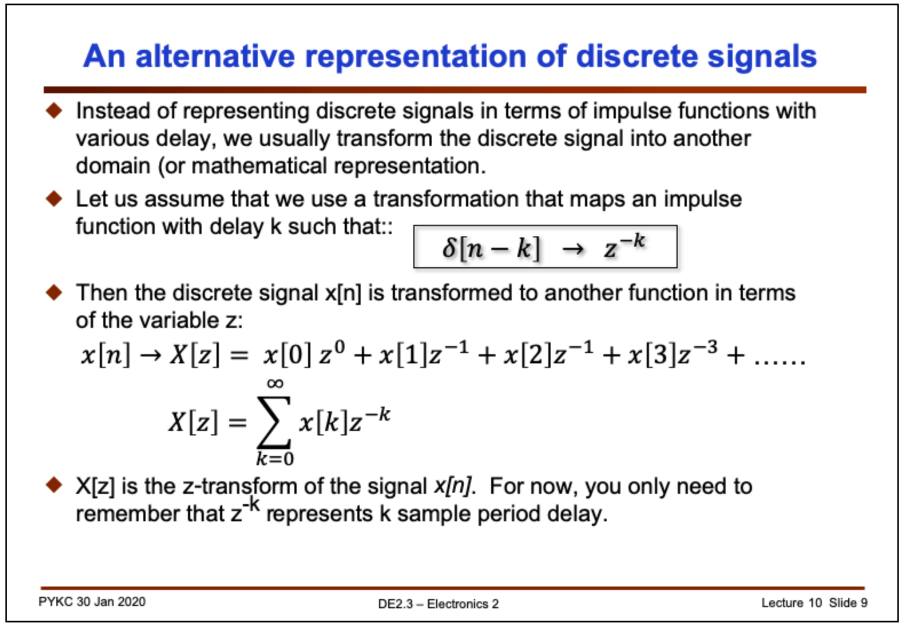
$$E\{x_i[n]\} = \sum_{k=0}^{K-1} |x[i+k]|^2$$

To compute the energy of a discrete signal, we only need to sum the square of each of the sample as shown here:

$$E\{x[n]\} = \sum_{k=0}^{\infty} |x[k]|^2$$

Often, we are not interested in the TOTAL ENERGY of a signal. A more useful measure is the energy of the signal over a finite window (over K samples). This is called instantaneous energy and is defined as:

$$E\{x_i[n]\} = \sum_{k=0}^{K-1} |x[i+k]|^2$$



An alternative representation of discrete signals

- Instead of representing discrete signals in terms of impulse functions with various delay, we usually transform the discrete signal into another domain (or mathematical representation.
- Let us assume that we use a transformation that maps an impulse function with delay k such that:: $\boxed{\delta[n-k] \;\rightarrow\; z^{-k}}$
- Then the discrete signal x[n] is transformed to another function in terms of the variable z:

$$x[n] \rightarrow X[z] = x[0]\,z^{0} + x[1]z^{-1} + x[2]z^{-1} + x[3]z^{-3} + \ldots\ldots$$

$$X[z] = \sum_{k=0}^{\infty} x[k]z^{-k}$$

- X[z] is the z-transform of the signal *x[n]*. For now, you only need to remember that z^{-k} represents k sample period delay.

Now I want to introduce a new concept known as z-transform. This is yet another useful transforms in signals and systems, and is used for handling (mathematically) discrete time signals. However, I want you to take the contents of this slide on faith. I will show you in a later lecture how the z-transform is derived from, and relate to), the Laplace transform.

For now, I want you to accept that if you take a unit impulse δ[n] and delay it by k samples, then the delayed version can be represented (in the z-domain) as z^{-k}.

The z-domain version of x[n] is written as X[z], similar to Laplace where we use uppercase X and the variable is in z (not n).

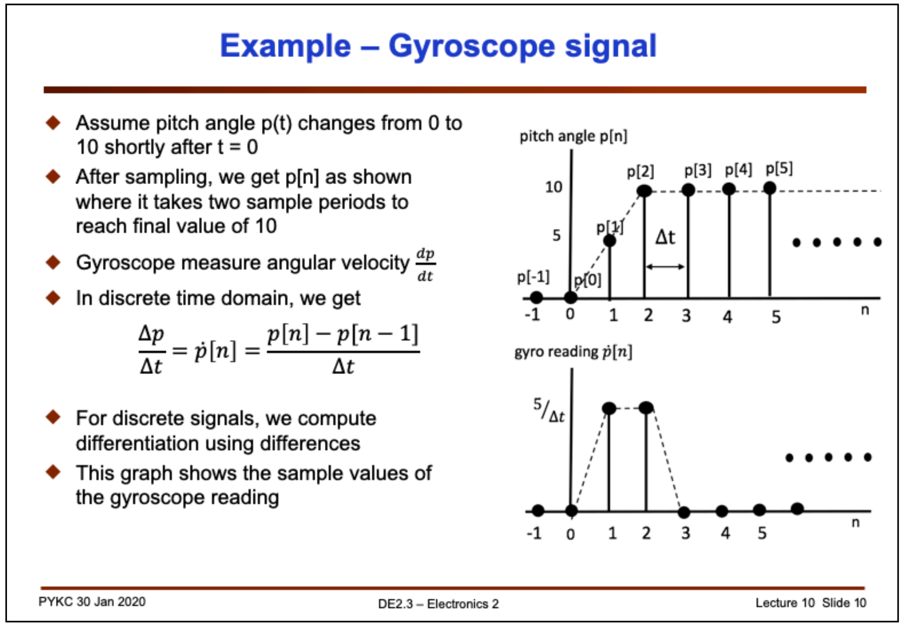
Now we have a new representation in the z-domain for the signal x[n] as:

$$x[n] \rightarrow X[z] = x[0]\,z^{0} + x[1]z^{-1} + x[2]z^{-1} + x[3]z^{-3} + \ldots\ldots$$ im

$$X[z] = \sum_{k=0}^{\infty} x[k]z^{-k}$$

The most imporatnt takeaway message here is, in the z-domain, we represent a k sample interval DELAY operation by the term z^{-k}.

Why we do this and why is this useful? I will explain this in detail in a later lecture.

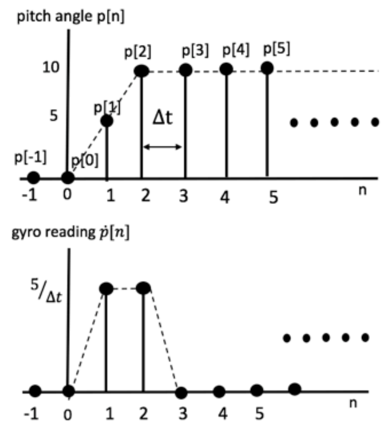


Example – Gyroscope signal

- Assume pitch angle p(t) changes from 0 to 10 shortly after t = 0
- After sampling, we get p[n] as shown where it takes two sample periods to reach final value of 10
- Gyroscope measure angular velocity $\frac{dp}{dt}$
- In discrete time domain, we get

$$\frac{\Delta p}{\Delta t} = \dot{p}[n] = \frac{p[n]-p[n-1]}{\Delta t}$$

- For discrete signals, we compute differentiation using differences
- This graph shows the sample values of the gyroscope reading

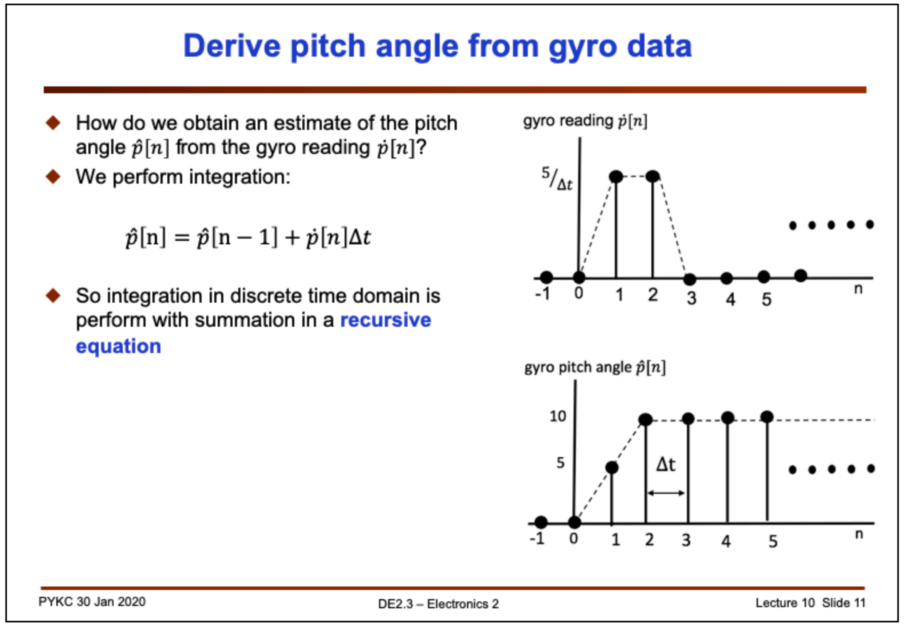


Let's apply what you have learned so far to some real signals you used in the lab. With the IMU, we obtained an estimate of the pitch angle (y-axis rotation) using the gyroscope. Assume the pitch of the Pyboard changes from 0 degree at t=0 to 10 degrees. This is effectively a step function.

However, the pitch angle is sampled with a period Δt. Let us assume that it takes two sample periods for the pitch angle to reach 10. We get the plot of discrete values from the IMU (via the I2C interface) as shown in the plot here.

This angle change is measured by the gyroscope as an angular velocity. How does one represent derivative (1st order differentiation) in discrete time (as we do within a processor)? Answer: we use difference equation or take the difference between successive samples:

$$\frac{\Delta p}{\Delta t} = \dot{p}[n] = \frac{p[n]-p[n-1]}{\Delta t}$$



Derive pitch angle from gyro data

- How do we obtain an estimate of the pitch angle $\hat{p}[n]$ from the gyro reading $\dot{p}[n]$?
- We perform integration:

$$\hat{p}[\mathrm{n}] = \hat{p}[\mathrm{n}-1] + \dot{p}[n]\Delta t$$

- So integration in discrete time domain is perform with summation in a **recursive equation**

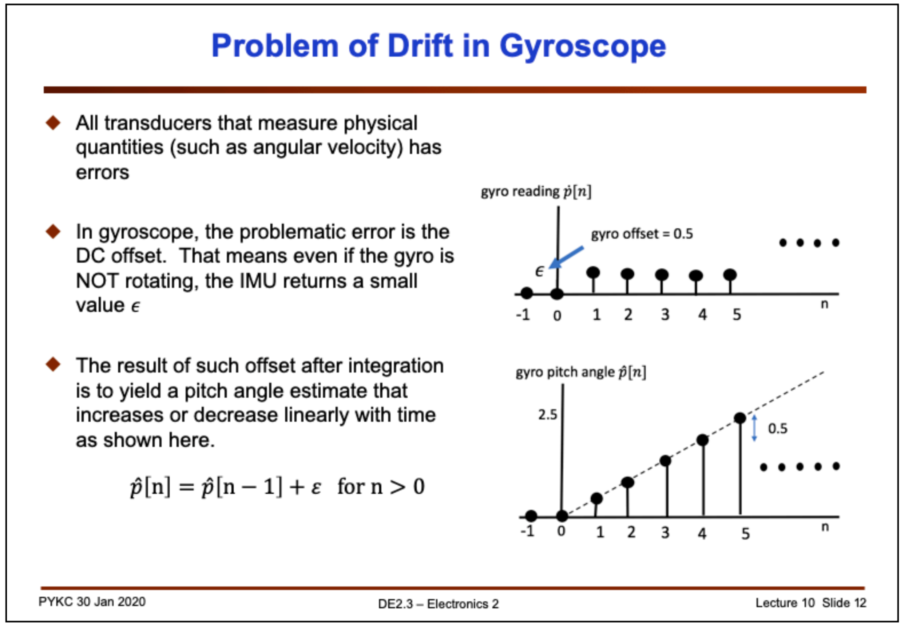
gyro reading $\dot{p}[n]$
$^5/_{\Delta t}$
-1 0 1 2 3 4 5 n
gyro pitch angle $\hat{p}[n]$
10
5
Δt
-1 0 1 2 3 4 5 n

With the gyro, what is being measure is $\dot{p}[n]$. From this, we need to derive an estimate of the pitch angle $\hat{p}[n]$ through integration.

In discrete domain, integration is achieved by accumulating the input samples:

$$\hat{p}[n] = \hat{p}[n-1] + \dot{p}[n]\Delta t$$

This equation is said to be **recursive** because the future value is of $\hat{p}$ is computed from previous value(s) of $\hat{p}$.



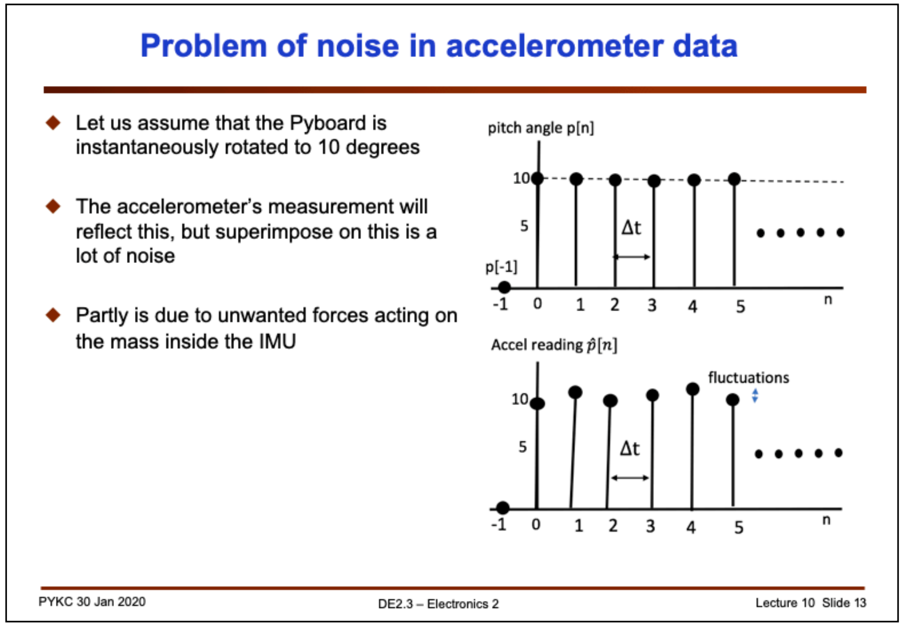
Problem of Drift in Gyroscope

- All transducers that measure physical quantities (such as angular velocity) has errors
- In gyroscope, the problematic error is the DC offset. That means even if the gyro is NOT rotating, the IMU returns a small value ϵ
- The result of such offset after integration is to yield a pitch angle estimate that increases or decrease linearly with time as shown here.

$$\hat{p}[\mathrm{n}] = \hat{p}[\mathrm{n}-1] + \varepsilon \quad \text{for n} > 0$$

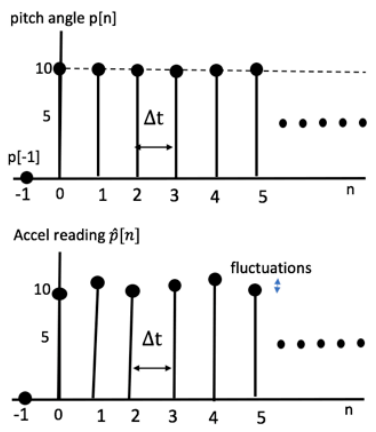
Unfortunately all gyroscope provide measurement for $\dot{p}[n]$ which has a DC offset error. Such an error, after integration, get accumulated over time. This manifests itself as a linear rise or fall in the estimated pitch angle. For example with the above example, we get (assuming $\hat{p}[-1] = 0$):

n	0	1	2	3	4	5	6	7	8	9	10
$\hat{p}[n]$	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0	4.5	5.0



Problem of noise in accelerometer data

- Let us assume that the Pyboard is instantaneously rotated to 10 degrees
- The accelerometer's measurement will reflect this, but superimpose on this is a lot of noise
- Partly is due to unwanted forces acting on the mass inside the IMU



Estimating the pitch angle using the accelerometer instead of the gyroscope has its own problem. The accelerometer cannot distinguish acceleration due to tilting (i.e. gravitational force) or due to movement (i.e. vibration or motion).

Therefore instead of a nice step function, you will see a step function with high frequency noise added as shown here.

In the lecture next week, I will show you how to mitigate against these two undesirable effects using the complementary filter in Lab 3.