

Summary of previous lecture

- Faster varying signals require faster sine signal components (i.e., at higher frequency), thus have higher maximum frequency and thus higher bandwidth
- How to calculate Fourier transforms in computers:
DTFT \rightarrow DFT (FFT)
- Introduction to sampling of analogue signals and meaning of sampling in frequency (i.e., with DTFT)
- Using the FFT in Matlab

Digital signals

We live in a digital world

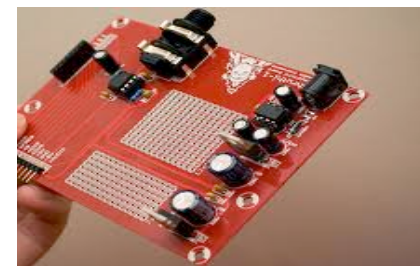
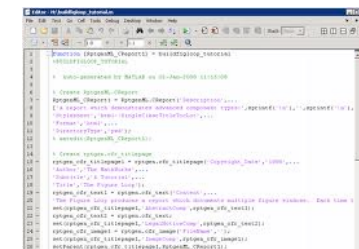
- We have all heard the word “digital”:
 - Almost all electronic devices we use are digital
 - Almost all information is processed digitally



But what does “digital” actually means??
Where does it come from??

Digital vs. Analogue processing

- Signal in nature are analogue:
 - they are continuous
 - and can take any value
- Since the advent of computers we have realized that we can store and process signals much better if we digitalize them and use computers to process them.
- Digital signals can be processed through software (easy to write)
- Analogue signals need hardware components to be processed

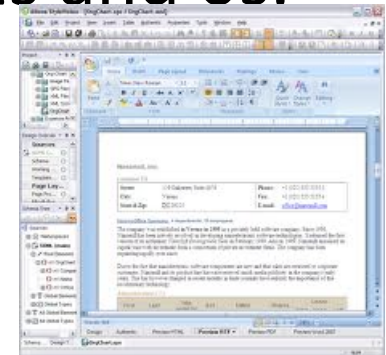
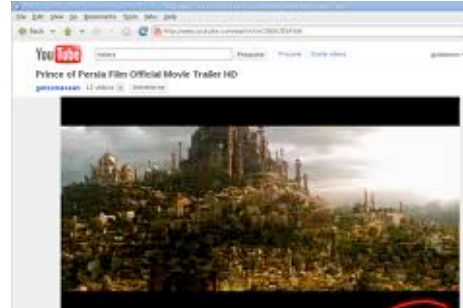


Digital vs. Analogue processing

- To write a digital filter, I can use software (C++, Matlab,...)
 - I can use software libraries, so I don't need to re-write all components
 - I can try to write software, compile it and see if it works...
- Processing signals in analogue mode requires analogue components (resistors, capacitors, inductors...)
 - I have to physically build a board to try it
 - I need to build every single block from scratch
 - It takes longer, it's more expensive and less flexible
- Think about flexibility:
 - A new hardware component needs to be built in a factory, needs to be shipped,... Plus factory only build components in thousands (or millions), they won't just build one for me...
 - A new software component can be written by any of you and disseminated through the Internet...

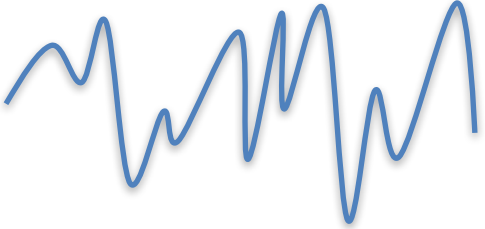
Digital signals and computers

- Every image, sound, text document, movie, etc. for a computer is just a sequence of 1s and 0s.
- Every processing of image, sound, text document, movie etc., for a computer is just a mathematical (logical) operation on a sequence of 1s and 0s.
- You see this:



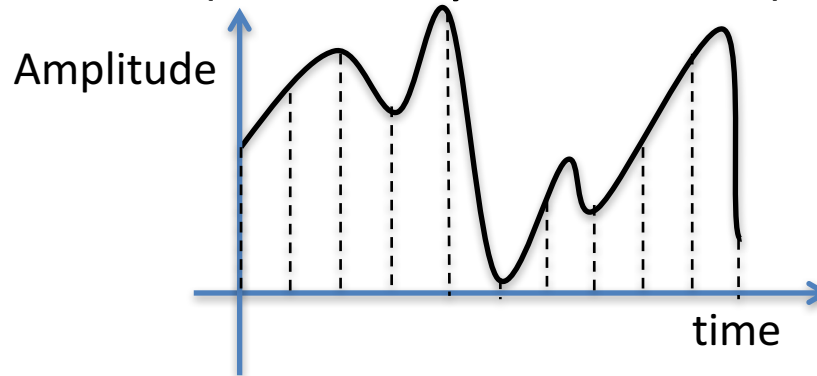
- Your computer sees this:
11001010011001101+
00111001001110100+...

Digital conversion of an analogue signal

- As we said all natural signal are analogue, continuous
- Digitalization means to transform
this:  into this: 110011011001
- Computers in fact work at the hardware level on digital binary format, a sequence of 1s and 0s

Sampling

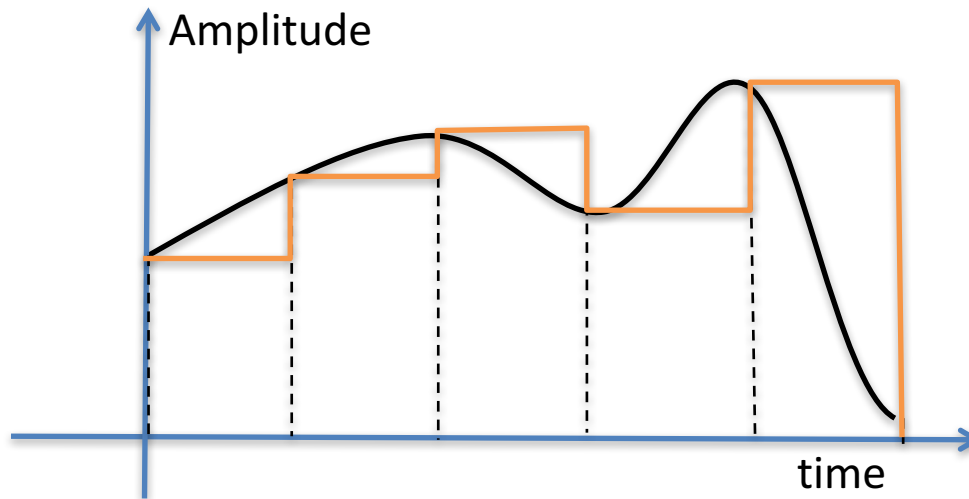
- The first step is to **sample** a signal, i.e. take a value at a certain moment in time
- I usually set a fixed sampling period. For example I could sample every 0.001 second (i.e., every millisecond)



- Sampling is usually expressed as sampling frequency, e.g. how many times do I sample in 1 second.
- $T = \frac{1}{f_s}$ where T is the sampling period, and f_s the sampling frequency
- So a period of 0.001 seconds corresponds to a sampling frequency of:
$$f_s = \frac{1}{T} = \frac{1}{0.001} = 1000 \text{ Hz} = 1 \text{ KHz}$$

Digitalization errors

- Consider sampling, we get the signal value at one point and we keep it until the next point
- The black one is the original signal, the orange one is the digital version



Do they look the same?

Does sampling introduce noise (errors)?

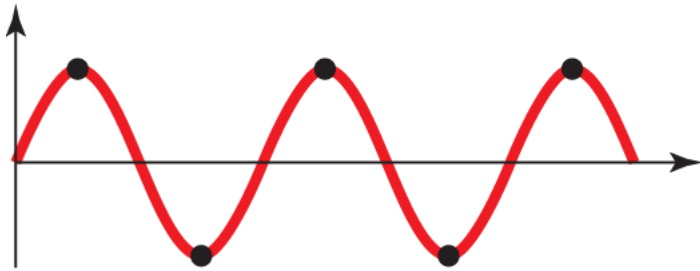
Nyquist-Shannon sampling theorem

- Believe it or not sampling does not introduce any noise as far as we select the right sampling frequency
- If my analogue signal has a maximum frequency f_{\max} :

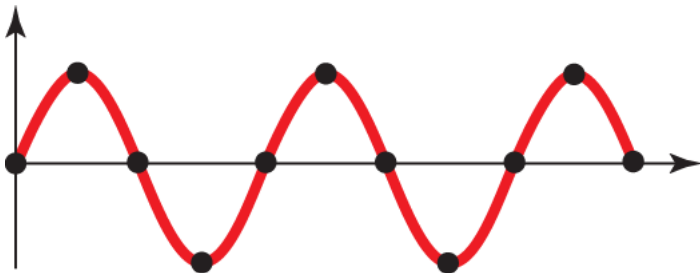
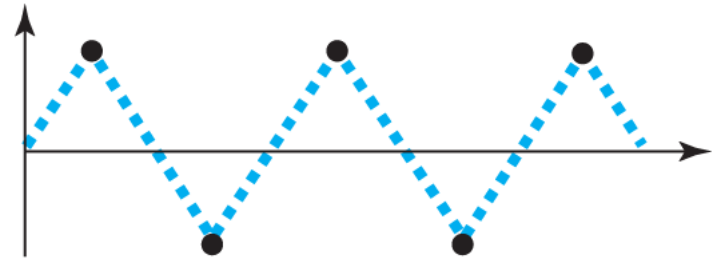
Sampling does not introduce any error if I use a sampling frequency:

$$f_s \geq 2 f_{\max}$$

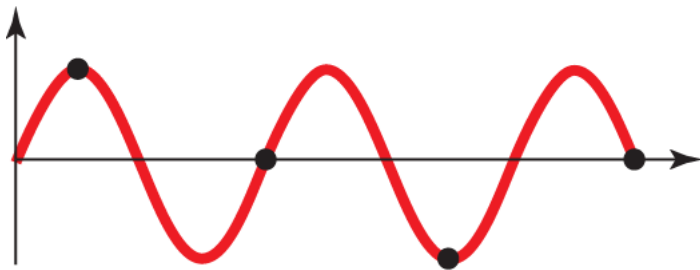
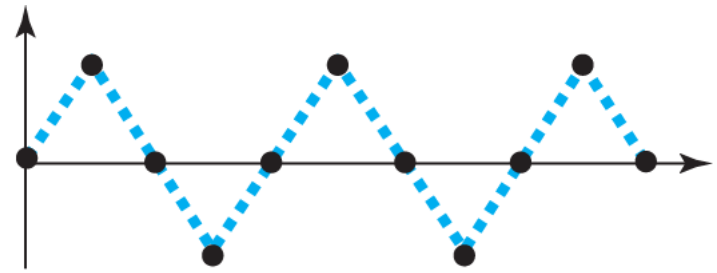
Recovery of a sine wave with different sampling rates.



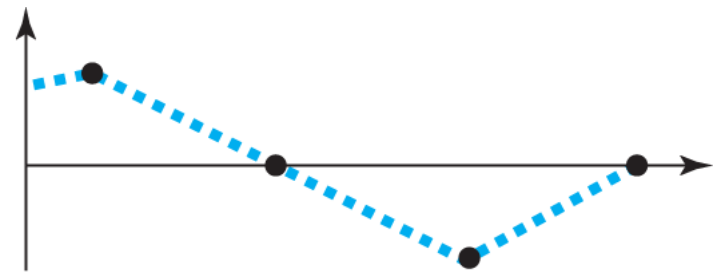
a. Nyquist rate sampling: $f_s = 2 f$



b. Oversampling: $f_s = 4 f$



c. Undersampling: $f_s = f$



Example of sampling

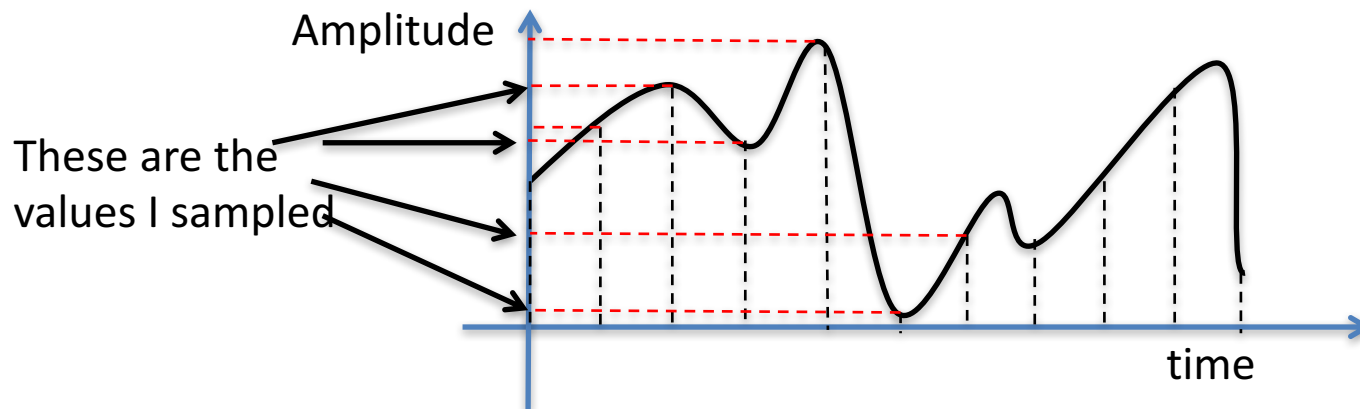
A complex baseband signal has a bandwidth of 200 kHz. What is the minimum sampling rate for this signal?

Solution

The bandwidth of a low-pass signal is between 0 and f , where f is the maximum frequency in the signal. Therefore, we can sample this signal at 2 times the highest frequency (200 kHz). The sampling rate is therefore 400,000 samples per second.

Quantisation

- Sampling will give me back a certain number of amplitude values (real numbers)



- Now I need to transform these real numbers into something that can be stored with a small number of bits (1s and 0s)

Quantization

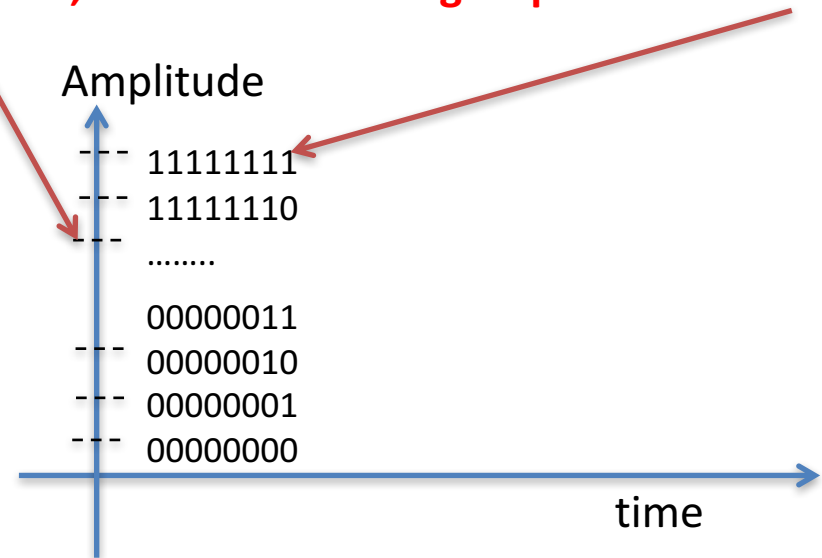
- Depending on how precise my measurement is, my sampled value could be very long, e.g., 6.4729476183950302...
- However I want to reduce the number of bits used to represent this number.
- For example I might want to fix the number of bits used to represent a sample to 8 bits (typical of fixed digital telephony)

8-bit Quantization

- With 8 bits (also known as 1 byte), I can represent 256 different amplitude levels
- If n is the number of bits I use then L is the number of different levels I get, following: $L = 2^n$
- So for 8 bits: $L = 2^8 = 256$

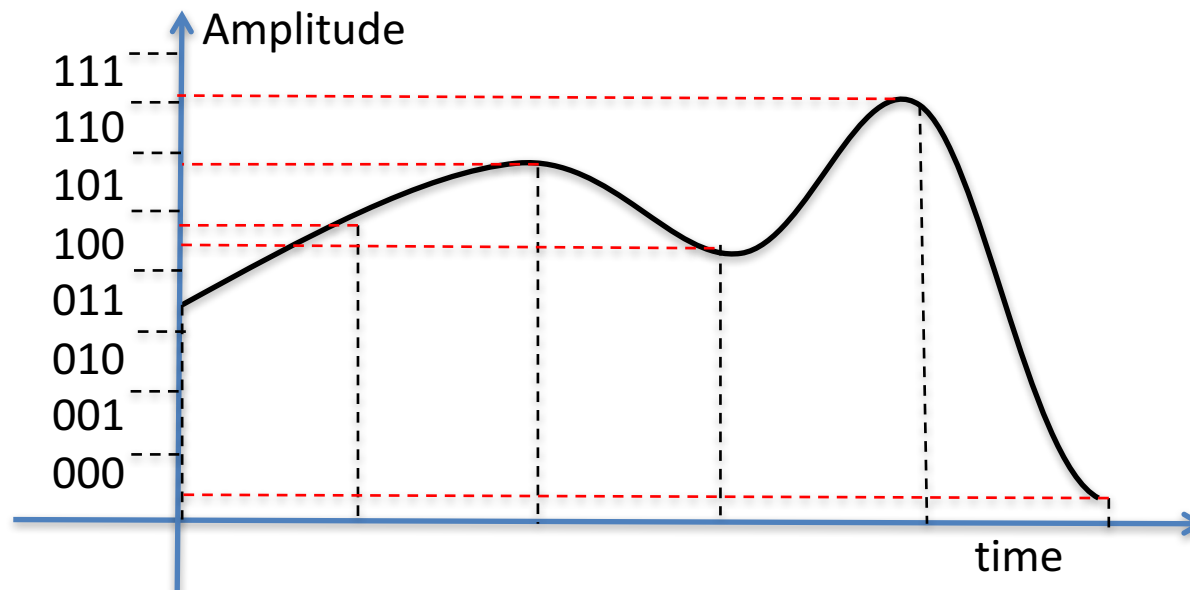
Note: 256 levels, are obtained using sequences of 8 bits

- What I do is take the y axis, and divide it into 256 equal levels and assign each of them a binary value



Quantization

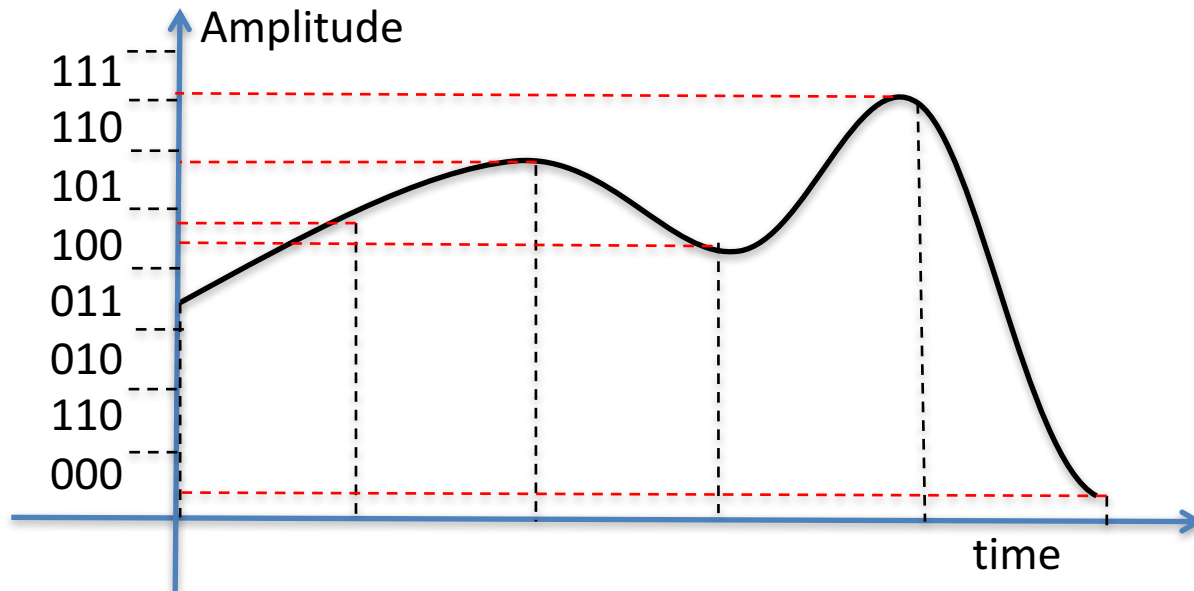
- For simplicity I consider in this plot only 8 levels (i.e., I use 3 bits)
- I divide the Amplitude in 8 levels, and associate each with a 3-bit label



- Then I check in which interval the sampled value falls and I represent it with the bits that classify the interval

Quantization example

- So for the signal in the example I would obtain the following series of samples:
- 011, 100, 101, 100, 111, 000



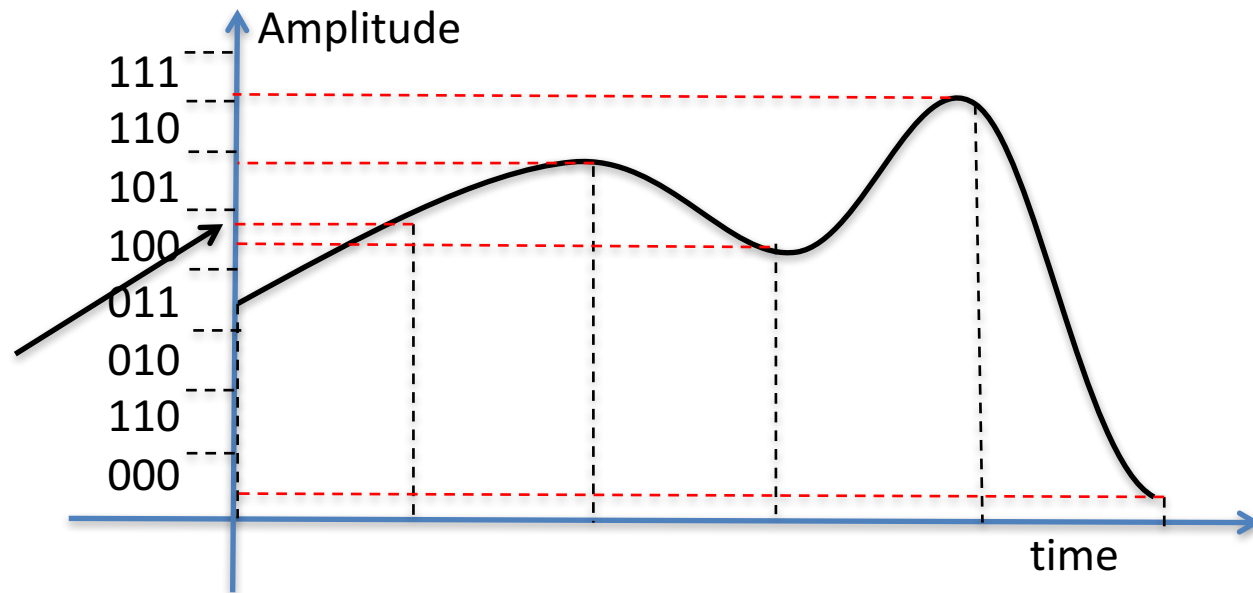
Quantization error

While, if $f_s \geq 2f_{\max}$ sampling does not introduce errors

quantization always introduces errors



These two values are different but get represented with the same digital value



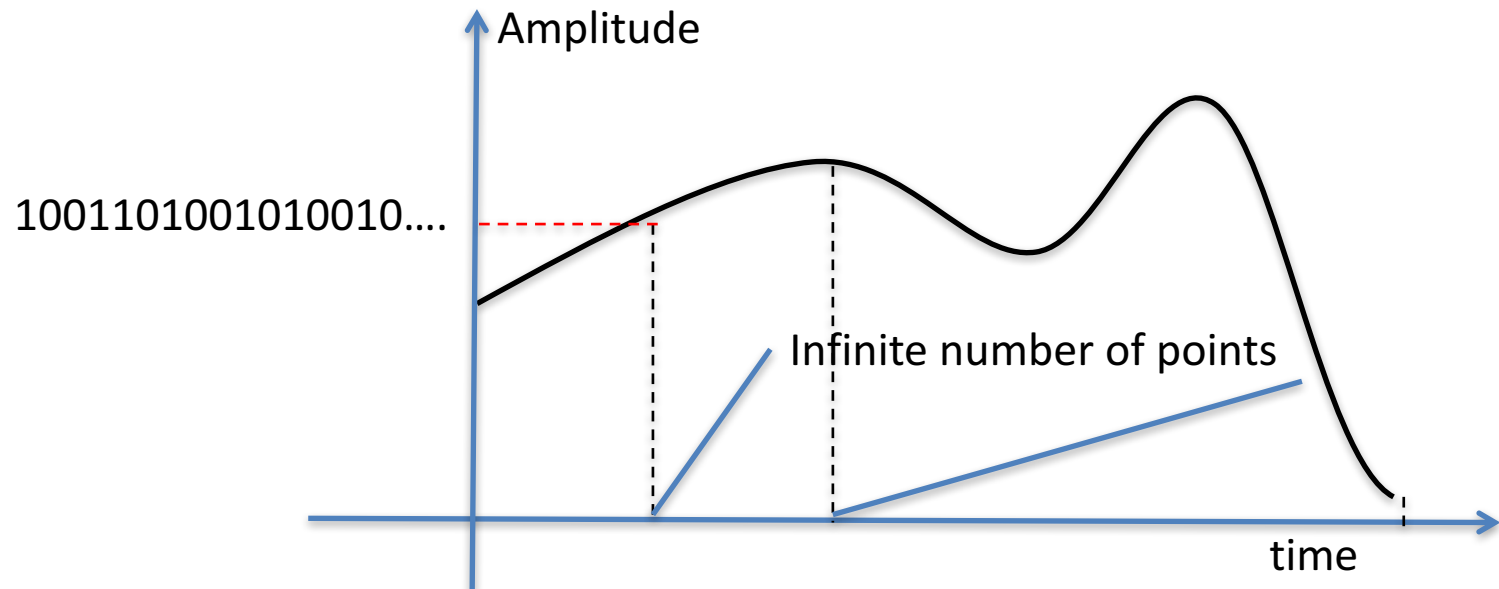
Every sampled value is represented with a value that is *slightly* different

Quantization error

- Quantization does introduce an error
 - This error can also be seen as noise, and thus called: quantization noise
 - Intuitively we can understand that if we use a higher number of bits then the distance between levels is smaller and we can define our samples more precisely.
- ➔ Although using a higher sampling frequency does not improve the quality of the signal
- Using a larger number of bits in the quantization does improve the quality of the signal
- This is the reason why a music CD uses a 16-bit quantisation, while the telephone uses only 8-bit quantisation.

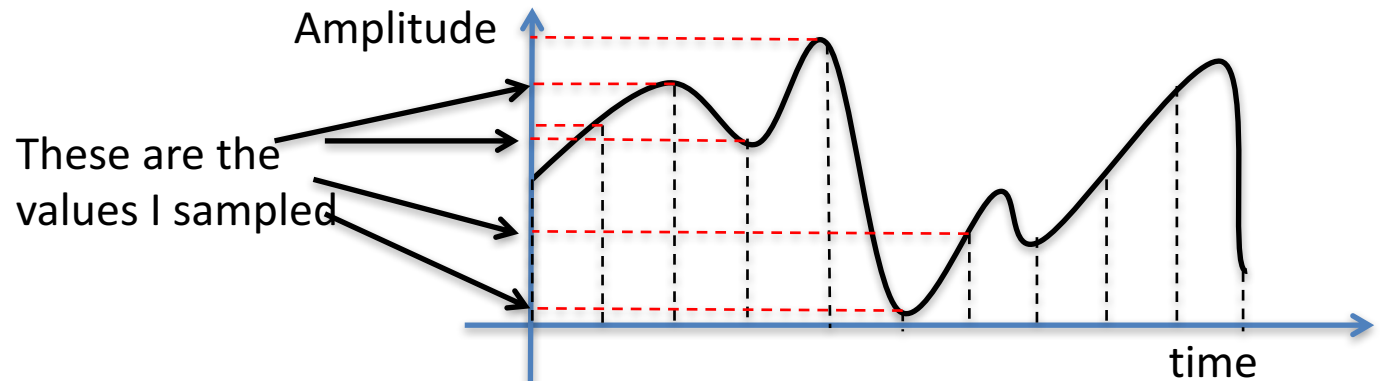
Summarising sampling and quantisation

- Analogue signals are continuous both in time and amplitude
 - ➔ Every time interval is made up of an infinite number of points
 - ➔ In order to represent with infinite precision each point in the amplitude axes it would take an infinite number of bits

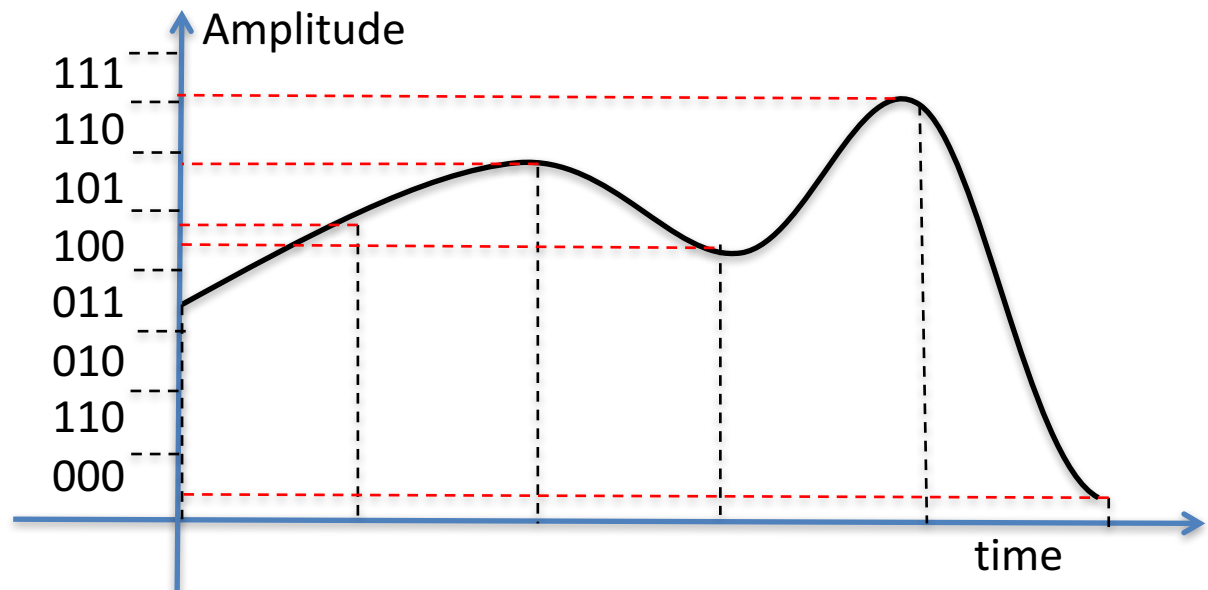


Summarising sampling and quantisation

- Sampling allows to take a finite number of points in time



- Quantisation allows to represent such points in the amplitude axes with a finite number of bits



Summarising sampling and quantisation

- Sampling does not introduce any error (nor noise, nor loss of information), provided we follow Nyquist-Shannon sampling theorem

$$f_s \geq 2f_{\max}$$

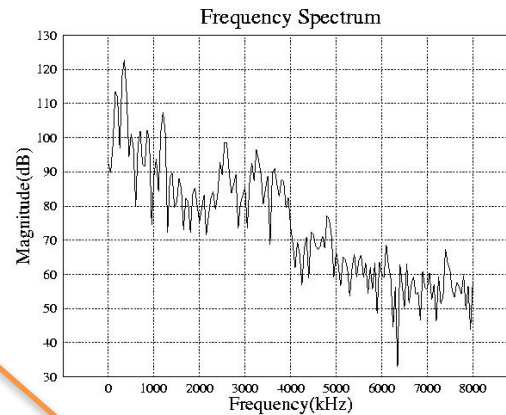
- However quantisation does introduce errors (or noise, or loss of information). The higher the number of bits I use for quantising the lower the error.

Why digital, if it implies loss of information?

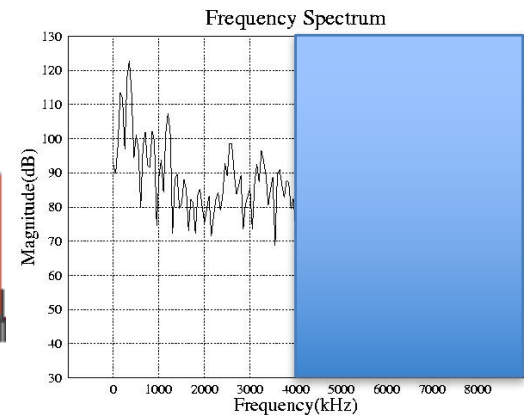
- Digital signals end up having better quality if you sample correctly and quantise with enough bits
 - Very hard to notice any loss of information if quantising at 16 bits or above.
 - The way analogue signals are stored and transmitted introduces noise. This is far worse than the noise introduced by quantisation (provided we use a suitable number of bits)
- ➔ Thus you end up getting a better experience with a digital signal
- ➔ In addition it has all the advantages mentioned at the beginning of this lecture

Digitalizing a telephone signal

- Telephone conversation:
 - Human voice has a spectrum up to about 10KHz



When it reaches the central office, the signal is filtered to 4KHz, and then digitalized



Digitalizing a telephone signal

- For Nyquist-Shannon we need to use a sampling frequency $\geq 2f_{\max}$
- For the telephone signal, after filtering, $f_{\max}=4\text{KHz}$
 \Rightarrow The sampling frequency is $2f_{\max}=8\text{KHz}$

What's the sampling period?

How often do we take a sample of the signal?

$$T = \frac{1}{f} = \frac{1}{8000} = 0.000125s = 125\mu s$$



Digitalizing a telephone signal

- When we quantize the phone signal we use 8 bits for each sample.
- The bit rate of the signal is
 $8 \times 8,000 \text{ KHz} = 64,000 \text{ bit/s}$ (bit per second)
 $= 64 \text{ Kbit/s}$

Now that we have a digital signal, we can introduce the concept of data rate, measured in bit per second.

Digitalizing music

- The best example is music on CDs
- Music has generally a higher bandwidth than voice. A recording including voice, guitars, drums,... can have a spectrum above 22KHz.
- In this case it's important not to cut everything above 4KHz, because:
 - Listening to music is not like having a phone conversation...
 - ...It's not just about recognizing the words, it's about having high-fidelity of the reproduction

Digitalizing music

- So where do we cut off?
- Remember that we need to cut off the signal somewhere because we need to know what is f_{\max} , to get our sampling frequency!
- Our ears are sensitive up to 22KHz (in average), so there is no reason for recording signals above 22KHz.
- So, $f_{\max}=22.05\text{KHz} \rightarrow f=44.1\text{KHz}$, $T=0.00002267\text{s}$
 $=22.67\mu\text{s}$

Digitalizing music

- 8 bit per sample is not very good for high-fidelity music.
- CD players use 16 bits per second
- Notice that this is much more, because the formula to indicate the number of quantization levels is exponential $L = 2^n$
- So for a phone call, with 8 bits, $L = 2^8 = 256$
- For a CD, with 16 bits $L = 2^{16} = 65,536$

Bit rate of CD music

- We have a sampling rate of 44.1KHz, 16 bits for each sample, and we have 2 channels (stereo recording)
- The bit rate is $44.1 \times 16 \times 2 = 1,411,200$ bit/s
 $= 1,411.2$ Kbit/s ≈ 1.411 Mbit/s
- This is about 22 times the data rate of a phone call!

Example 1

Assume we need to download text documents at the rate of 100 pages per second. The text has 24 characters per line and 80 lines per page. Each character is encoded with ASCII at 8 bits. What is the required bit rate of the channel?

Solution

We first need to find the number of bits required to encode the document.

- 1 page = 24×80 characters = $24 \times 80 \times 8$ bits = 15,360 bits
- 100 pages will require $15,360 \times 100 = 1,536,000$ bits
- Sending 100 pages per second requires a channel capable of $1,536,000$ bits/s = 1.536 Mbit/s

Example 2

What is the bit rate for high-definition TV (HDTV)?

Solution

HDTV uses 1920 by 1080 pixels per screen, and the screen is renewed 25 times per second. 24 bits represent one color pixel (8 bits per color).

- Each frame requires: $1920 \times 1080 \times 24 \text{ bits} = 49.7664 \text{ Mbits}$
- Each second we have $49.7664 \times 25 = 1.2442 \text{ Gbit/s}$

The TV stations reduce this rate to 8 to 15 Mbit/s using compression algorithms.

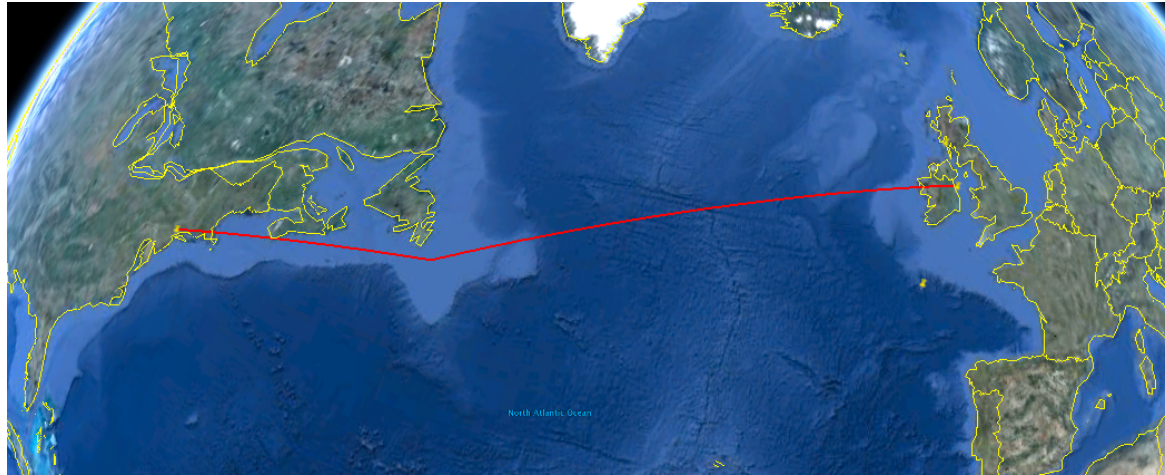
The Blu-ray disc uses a maximum of 40Mbit/s.

Data rate

- The data rate of a digital signal expresses the number of bits it is sending per seconds.
- A higher data rate gives the perception of a faster transmission of information.
- For example downloading a 100 MB file will take:
 - 13 minutes, 59 seconds at 1Mbit/s rate
 - 1 minutes, 24 seconds at 10 Mbit/s rate
- **However**: do not confuse propagation speed (latency) and data rate!
 - The signal will always travel at the same propagation speed, which is limited by the speed of light.
 - A higher data rate will allow to transmit a larger number of bits within a given time interval. But the time it will take for the first bit to reach its destination will be the same. Independently of the data rate.

Data rate vs. propagation speed

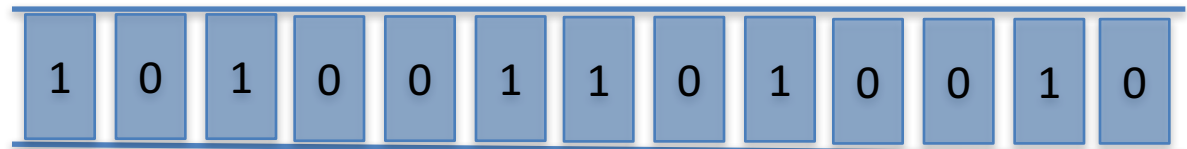
Imagine we have an optical fibre link from Dublin to New York, long 5,200 Km. The propagation time (latency) is about $5,200,000/200,000,000=26\text{ms}$.



A lower rate channel can be seen as a smaller pipe, where bits elongate to fit into it:



A higher rate channel can be seen as a larger pipe, where bits need shorter time slot each :

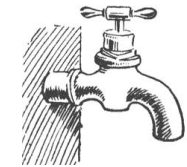


But the time it will take for the first bit to reach Dublin, once it leaves New York it's 26 ms, independently of the data rate.

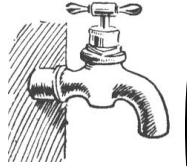
Analogy with water pipes

Imagine we have two water pipes, one with a diameter of 2 cm, another with diameter of 20 cm.

The pipes are both long 5 m and I want to fill in a 10 l bucket at the end of the pipes.



The small pipe carries 0.2 l per second



The large pipe carries 2 l per second



Imagine to apply the same pressure so that water travels at the same speed, say of 1m/s: for both pipes it will take 5 s before water comes out at the other end

On the small pipe the bucket will be filled in $10\text{ l} / 0.2\text{ l/s} = 50\text{ s} + 5\text{ s} = 55\text{ s}$

On the large pipe the bucket will be filled in $10\text{ l} / 2\text{ l/s} = 5\text{ s} + 5\text{ s} = 10\text{ s}$

Bytes, bits, rates

- A bit is a 1 or a 0 value, the basic unit of information.
- In computers bits are aggregated in groups of 8 to form a byte. For example one character encoded with ASCII occupies 8 bits, i.e. 1 byte.
- Usually we use bytes when we indicate data stored in a computer, e.g. size of a movie, and bits when we indicate transmitted data, e.g., the data rate of a streamed movie.
- When we have large number of bytes or bits we use a notation of Kilobytes, Megabytes,... Kilobits, Megabits... to simplify

Binary vs. metric notation

- Historically stored data uses a binary notation, while transmitted data a metric notation

Stored data

Notice the difference between the Ki, Mi notation used for binary and the K, M, used for decimal

Bytes Notation	Decimal notation	Bytes (decimal)	Binary notation	Bytes (binary)	Bytes (binary)
1 Byte	1 B	1 B	1 B	2^1 B	1 B
1 Kilobyte	1 KB	1000 B	1024 B	2^{10} B	1024 B
1 Megabyte	1 MB	1000 KB	1024 KiB	2^{20} B	1,048,576 B
1 Gigabyte	1 GB	1000 MB	1024 MiB	2^{30} B	1,073,741,824 B
1 Terabyte	1 TB	1000 GB	1024 GiB	2^{40} B	1,099,511,627,776 B

Transmitted data

Bits Notation	Brief notation	Bits per second
1 bit per second	1 bit/s	1 bit/s
1 Kilobit per second	1 Kbit/s	1000 bits/s
1 Megabit per second	1 Mbit/s	1×10^6 bits/s
1 Gigabit per second	1 Gbit/s	1×10^9 bits/s
1 Terabit per second	1 Tbit/s	1×10^{12} bits/s