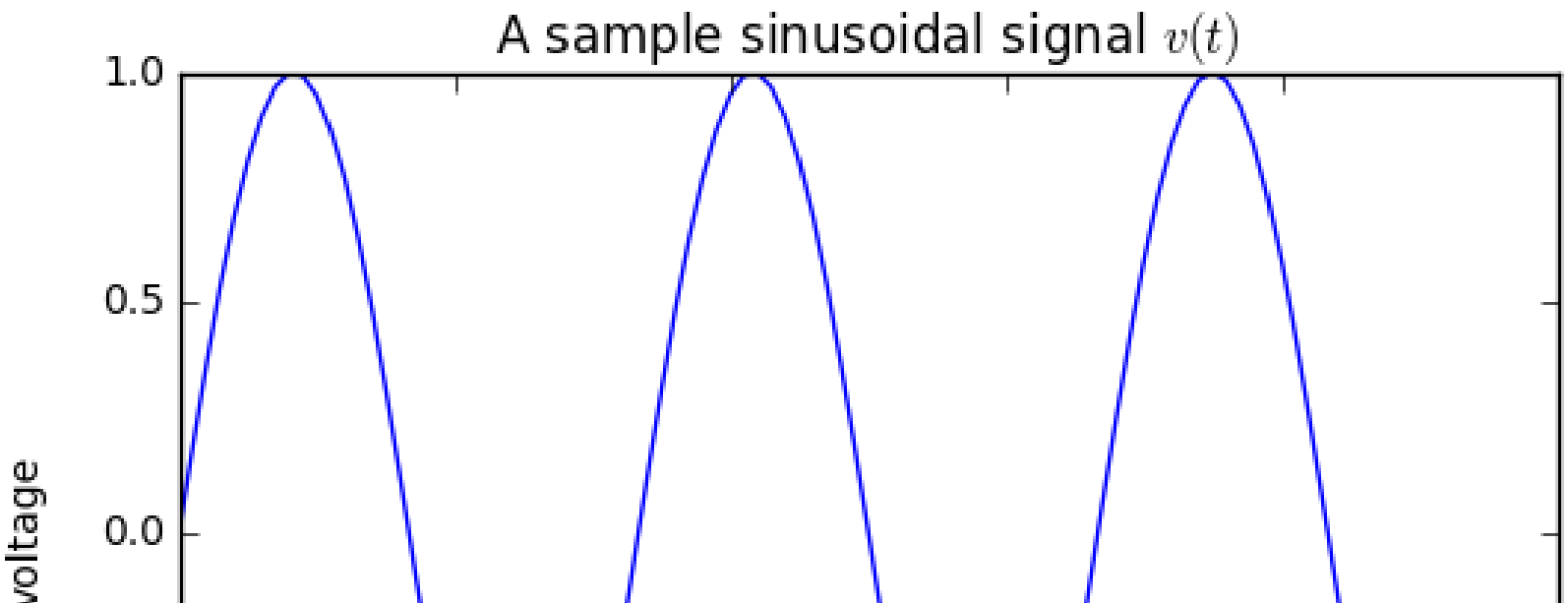
I. Sampling of Analog Signals

I.1 Signals

A **signal** is a measurable quantity which varies in time, space or some other variable. Examples: - a voltage which varies in time (1D voltage signal) - sound pressure which varies in time (sound signal) - intensity of light which varies across a photo (2D image) Signals are usually represented as mathematical functions, e.g. $v(t)$.

```
%matplotlib inline
import matplotlib.pyplot as plt, numpy as np, math
t = np.arange(0,100)                    # this means 0,1,2,...100
v = np.sin(2*math.pi*0.03*t)            # sin(2*pi*f*t)
plt.plot(t,v)                           # plotting
plt.xlabel('time')
plt.ylabel('voltage')
plt.title('A sample sinusoidal signal $v(t)$')
```

```
<matplotlib.text.Text at 0xa582c10c>
```



I.2 Discrete and analog frequency

A signal is called **periodic** if its values repeat themselves after a certain time (known as **period**).
For an analog signal:

$$x(t) = x(t + T), \forall t$$

For a discrete signal:

$$y[n] = y[n + N]), \forall t$$

The **fundamental period** of a signal is the minimum value of T or N. Note that multiples of T or N are also periods ($2T$, $3T$ etc), but we are usually interested only in the fundamental period. Therefore, from now on, by *period* we will mean the fundamental period.
Note that for analog signals the period has unit *seconds*, but for digital signals the period is *adimensional*. This is because T is time, but N is just a number and therefore it does not have a unit attached.
The frequency of a signal is defined as the inverse of the period.
For analog signals:

$$F = \frac{1}{T},$$

and the unit is:

$$[F] = \frac{1}{s} = Hz.$$

For discrete signals:

$$f = \frac{1}{N},$$

and it has no unit, since N has no unit also.
We will frequencies of analog signals with F and frequencies of discrete signals with f. Thus, we can say that an analog signal has a frequency of $F = 0.1Hz$, but a discrete signal has a frequency of $f = 0.1$

I.3 Sampling of analog signals

Discrete signals are usually obtained from analog signals through the process of **sampling**.

Sampling means taking the values from the analog signal at certain discrete moments of time (usually periodic).

The time between two samples is the **sampling period** T_s. The corresponding frequency is the **sampling frequency** $F_s = \frac{1}{T_s}$.

```
%matplotlib inline
import matplotlib.pyplot as plt, numpy as np, math
t = np.arange(0,100)                    #
xa = np.sin(2*math.pi*0.03*t)           # analog signal
Ts = 8                                  # Sampling period Ts=10, sampling freq Fs =
td = t[0:101:Ts]                        # go from 0 to 100 with step=Ts, i.e. 0,10,
xd = xa[td]
print td
plt.plot(t,xa)                           # plotting
plt.stem(td,xd,'--r')
plt.xlabel('time')
plt.ylabel('voltage')
plt.title('A sample sinusoidal signal $v(t)$')
```

```
[ 0  8 16 24 32 40 48 56 64 72 80 88 96]
```

```
<matplotlib.text.Text at 0xa26f580c>
```

I.4 Signal quantization and coding

After sampling a real-valued analog signal, the samples can have any real value.
Quantization is the process of truncating a value to a limited set of predefined values (**quantization levels**).
A rigurous presentation is outside the scope of these lectures. We only give some illustrative examples.
For example, the grade of a student can be any real value between 0 and 10 (say, 8.75). Due to administrative reasons, the value needs to be truncated to one of the integer numbers 1, 2... 10. This is quantization of the value, and the quantization levels are the possible integer grades.
Another example: we sample an analog voltage signal with possible values between 0V and 10V. We need to store the value on one byte (8 bits). With 8 bits we have $2^8 = 256$ different possible values, which can go from 0 to 10. Therefore we divide the interval $[0 - 10]$ in 255 equal sub-intervals of size $\frac{10}{255} \approx 0.039V$, so that we have 256 quantization levels, corresponding to the 256 possible numeric values: $0 = 0V$, $1 \approx 0.039V$, $2 \approx 0.078V$, $\ldots 256 = 10V$. The values of the samples are rounded to the closest quantization level.
The difference between quantized value and the original value is the **quantization error**.
Quantization can be done via **truncating** (the chosen cuantization level is the one immediately smaller than the value) or via **rounding** (the chosen cuantization level is the closest to the value, either smaller or larger). Truncating is simpler, but rounding is usually preferred because of smaller quantization errors.
Coding is the process of converting a quantized value in binary form, such that it is usable by the processor or microcontroller in a digital system.

I.5 A/D and D/A Conversion

Typically, the whole process of sampling, quantization and coding is done by a single circuit known as **Analog to Digital Converter (ADC)**.
The inverse operation of reconstructing an analog signal from numeric samples is done by a **Digital to Analig Converter (DAC)** circuit. Usually reconstruction is not done based on the ideal reconstruction equation presented above, which is too compex, but with simpler empiric approach.