# Module IN3031 / INM378 Digital Signal Processing and Audio Programming

Johan Pauwels
johan.pauwels@city.ac.uk
based on slides by Tillman Weyde

# Signal Correlation
# Fourier Transform

# Dot Product

- The **dot product** (also called **inner** or **scalar product**) is the basic way of measuring the **correlation** (similarity) of two signals by adding their products at each sampled time

$$dot(x,y)=\sum_{t=0}^{N-1} x[t]\cdot y[t]$$

- The basic intuition:
**similar sample values** x[t], y[t] give **greater dot product**
  - x[t] and y[t] big, same sign → dot(x,y) big positive
  - x[t] and y[t] big, different sign → dot(x,y) big negative
  - x[t] and y[t] small abs values → dot(x,y) small abs

# Dot Product and Correlation

- The **dot product** between a **signal and itself**, the **autocorrelation** at lag 0, is the **energy** of the signal,
- The term **correlation** is used for **different variants** of **similarity measures** in several areas of mathematics and applications

# Autocorrelation

- **Autocorrelation** measures the similarity of a **signal** with **itself** at a certain **time** or space **lag**

$$autocorr(x,k) = \sum_{t=0}^{N-1} x[t] \cdot x[t+k]$$

- The **autocorrelation** of a **signal at lag 0** is the **energy** of the signal. Python: np.*dot(x,x)*
- Auto-Correlation is useful for **detecting periodicities**.
- Python: np.*correlate(x,x)*

# Cross-Correlation

- **Cross-correlation** measures the similarity of two signals at a time lag *k*:

$$xcorr(x, y, k) = \sum_{t=0}^{N-1} x[t] \cdot y[t+k]$$

- The cross-correlation between a signal and itself is the **auto-correlation**
- Cross-Correlation is useful for **measuring delays**.
- Values outside the signal time range are assumed as 0.
- Python: np.*correlate(x, y, mode='full')* calculates the result for all values of *k*

# Energy of Added Signals

- **Example**: **Convert** a **stereo** signal **to mono by adding** the two signals (possibly dividing by 2 to avoid clipping).
- The **energy** of the **resulting signal** depends on the correlation of the signal:

$$\sum (x[n]+y[n])^2 = \sum (x[n]^2 + 2\cdot x[n]\cdot y[n] + y[n]^2)$$
$$= \sum x[n]^2 + \sum y[n]^2 + 2\,dot(x,y)$$

- If x = y, we have $(2x[n])^2 = \mathbf{4\,x[n]^2} = x[n]^2 + x[n]^2 + 2\cdot x[n]\cdot x[n]$
- if x = -y, it's $(x[n]-x[n])^2 = \mathbf{0} = x[n]^2 + x[n]^2 - 2\,x[n]\cdot x[n]$ (total cancellation)

# Cross-Covariance

- **mean-removed** cross correlation
- Is the same as **correlation**, but removes the mean (DC offset) of both signals before processing
- Example:
  - x = [10, 0, -10, 0, 10]  -> mean(x) = 2
  - Remove mean: y = x – mean(x) (per sample)
  - y = [ 8, -2, -12, -2, 8 ] -> mean(y) = 0

# Correlation Coefficient

$$\sum (x[n]+y[n])^2 = \sum (x[n]^2 + 2x[n]\cdot y[n] + y[n]^2)$$
$$= \sum x[n]^2 + \sum y[n]^2 + 2\,dot(x,y)$$

The **correlation coefficient** ρ (Greek letter 'rho') is the ratio of the correlation energy to the geometric mean of the mean-free signal energies:

$$\rho = \frac{dot(x,y)}{\sqrt{\sum x[n]^2 \cdot \sum y[n]^2}}$$

ρ = 1 means $x = y \cdot z$ ($z>0$)

ρ = -1 means $x = y \cdot z$ ($z<0$)

# Correlation Coefficient

- The correlation coefficient ρ measures the **similarity** of signals *x, y* **independent of scaling**.
- **If** both signals are **mean-free** (i.e. mean(s) = 0) and have the **same energy**:
  - ρ = 1 means x = *y*, correlation energy is equal to sum of energies of *x* and *y*, energy(*x*+*y*) = 4·energy(x)
  - ρ = -1 means x = -*y* , correlation energy is the negative of the sum of energies of both signals, energy(*x*+*y*)=0
  - ρ = 0 means that signals are not correlated, i.e. energy(*x*+*y*) = energy(*x*) + energy(*y*)

# Appraising Mono Compatibility

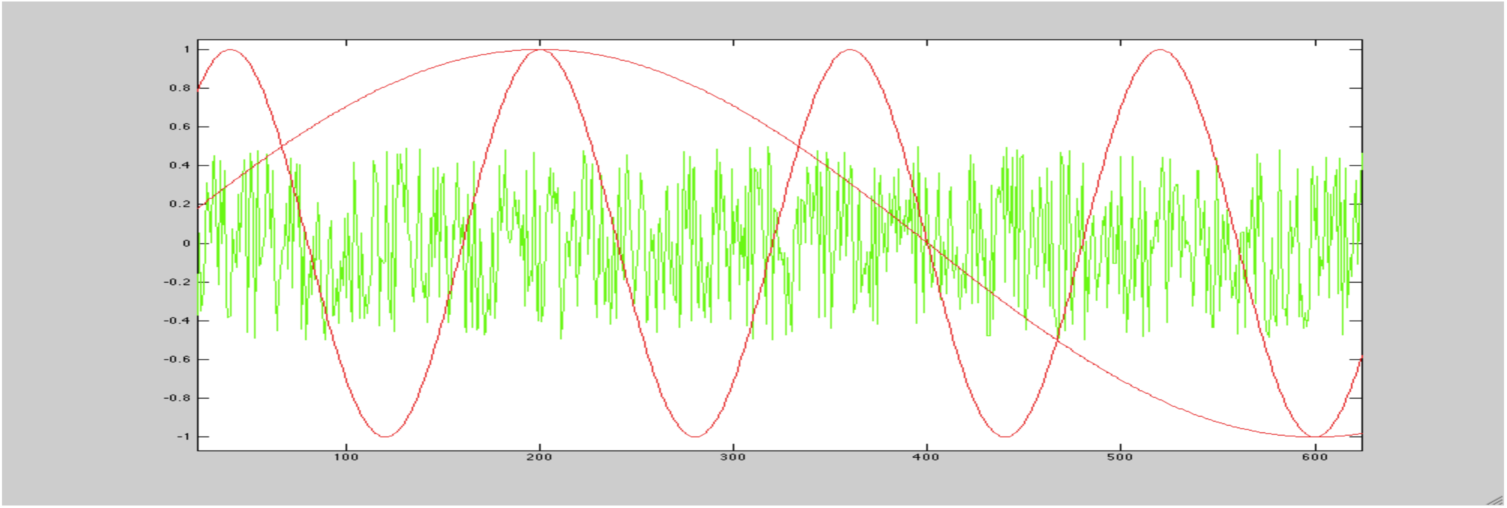
- Appraise the **Mono compatibility** of this stereo signal:
  $r$ =[ 2, 1, 0,-1,-2], $l$ =[1, 2, 0,-2,-1]
- **Correlation** is 2 + 2 + 0 + 2 + 2 = 8
- $r$ and $l$ have mean 0 (**DC free**)
- **Energy**: ($r$) 4+1+0+1+4 = 10, ($l$) 1+4+0+4+1 = 10
- **Corr. coefficient**: ρ = 8/ sqrt(10·10) = 8/10 = 0.8
- This indicates good mono compatibility (close to 1).

# Frequency Analysis

# Finding Frequency Components

For a periodic (i.e. repeating) function we can

- find signal components using **dot product**
- use **sine functions** as **building blocks**



# Dot Product with a Sine

Dot product of a signal ***x*** with sine wave of frequency ***f***, ***sf*** := sin(2*pi*f/Fs):

$$dot(x, sf) = \sum_{t=0}^{N-1} x[t] \cdot \sin[2 * pi * f / FS * t]$$

- *dot(x,sf)* tells us **how similar *x*** is to ***sf***
- **Interpretation**: **how much** of ***sf*** is contained **in *x***

# Which Frequencies to Try

- We use **frequencies** depending on the **period p** (i.e. the time after which the signal repeats)

  $f_1 = 1/p$

  $f_n = n/p$

$$s_1 = \sin(2\pi f_1 t)$$
$$s_2 = \sin(2\pi f_2 t)$$
$$s_3 = \sin(2\pi f_3 t)$$
$$s_4 = \sin(2\pi f_4 t)$$
$$\ldots$$

# Correlations of Sines

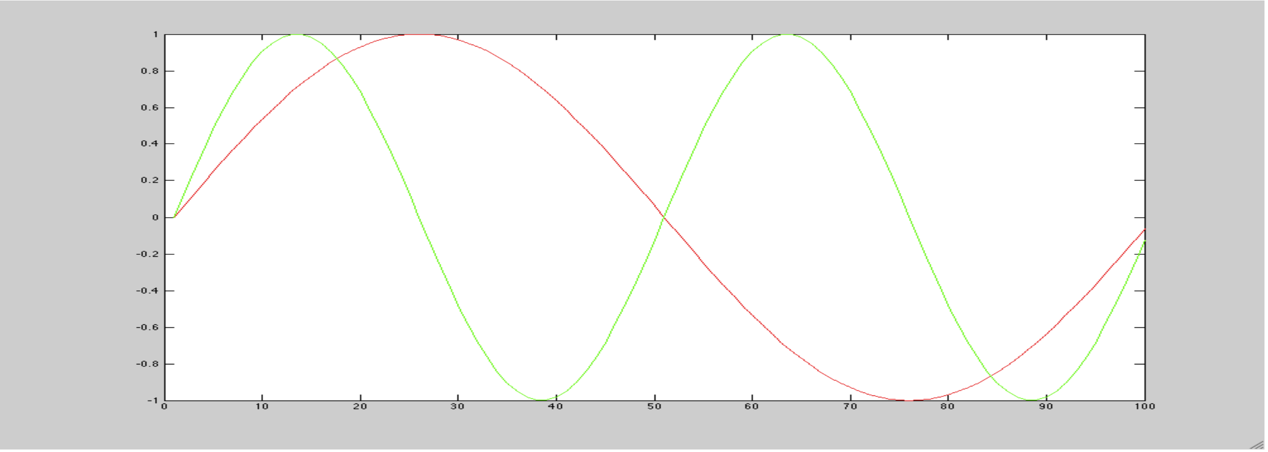
Different **sines in our series** have **zero correlation**

$$dot(sf_1, sf_2) = \sum_{t=0}^{N-1} sf_1[t] \cdot sf_2[t] = 0$$

$$\text{where } sf_1[t] = \sin[2 \cdot pi \cdot f_1 / Fs \cdot t]$$

$$\text{and } sf_2[t] = \sin[2 \cdot pi \cdot f_2 / Fs \cdot t] \, and \, N = p \cdot Fs$$

- so we are **separating frequencies** properly



# Offsets on the Time Axis

**Sine** waves of **same frequency** but with **offset** on the **time axis** (**sinusoids**) can produce **different correlations**:

$$dot(sf[t], sf[t+k]) = \sum_{t=0}^{N-1} sf[t] \cdot sf[t+k]$$
$$\text{where } 0 < k < f/Fs$$
$$\text{and } sf[t] = \sin[2 \cdot pi \cdot f/FS \cdot t]$$

If **offset** (in samples) k = 0.5 *f/Fs* (half cycle length)

**correlation switches sign**

$$dot(sf[t], sf[t+.5\, f/Fs]) = -dot(sf[t], sf[t])$$
$$\text{because } \sin(x) = -\sin(x+\pi)$$

# Offsets on the Time Axis (2)

**Sines** at **offsets 1/2 π** and **3/2 π** can be represented as **cosines**:

$$\cos(x) = \sin(x + \pi/2)$$
$$-\cos(x) = \sin(x + 3\pi/2)$$

**Sine** plus **cosine** (with suitable **a,b**) can represent any offset:

$$\forall \phi \exists a, b \quad \textit{such that}$$
$$\sin(x+\phi) = a\sin(x) + b\cos(x)$$

***a,b*** can be **determined by the dot product**:

$$a = dot(f, \sin) \cdot 2/N,\ b = dot(f, \cos) \cdot 2/N$$

# Fourier Series (continuous)

- Famous insight by *Jean-Baptiste Joseph Fourier* in 1822:

  Any **periodic** analogue **signal** with period **p = 1/f** is represented uniquely and unambiguously by an infinite **series of sinusoids** with frequencies ***k*f**:

$$
\begin{aligned}
x(t) = b_0 \\
&+ a_1 \sin(2\pi 1\, f\, t) + b_1 \cos(2\pi 1\, f\, t) \\
&+ a_2 \sin(2\pi 2\, f\, t) + b_1 \cos(2\pi 2\, f\, t) \\
&+ a_3 \sin(2\pi 3\, f\, t) + b_3 \cos(2\pi 3\, f\, t) \\
&+ \ldots. \qquad = \sum_{k=0}^{\infty} a_k \sin(2\pi k\, t) + b_k \cos(2\pi k\, t)
\end{aligned}
$$

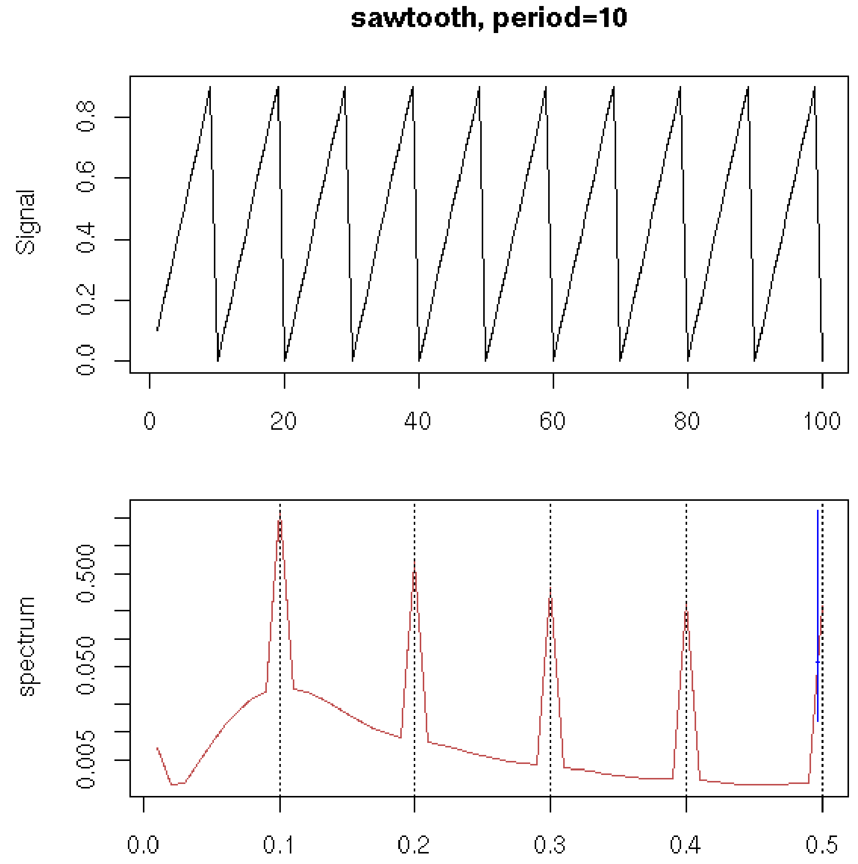
# Fourier Series (discrete)

- A **sampled signal** of length N can be represented uniquely and unambiguously by a finite **series of sinusoids**:

$$
\begin{aligned}
x[n] = a_0 & \\
& + a_1 \cos\left(2\pi \frac{n}{N} + \phi_1\right) \\
& + a_2 \cos\left(2\pi \frac{2n}{N} + \phi_2\right) \\
& + \ldots. \\
& + a_{N-1} \cos\left(2\pi \frac{(N-1)n}{N} + \phi_{N-1}\right) = \sum_{k=0}^{N-1} a_k \cos\left(2\pi \frac{kn}{N} + \phi_k\right)
\end{aligned}
$$

# Frequency Spectrum

- The **amplitude and phase distribution** of a signal **over frequencies** is called its **spectrum**.
- We will see now **how to calculate** the spectrum.



# Frequency Units

- As a **sine** or **cosine** function has a period of **2π** (in radians units, 2π is 360°), expressing frequencies in radians is often convenient.
- **analogue** frequencies
  - standard $f$ : **cycles/sec** (0 ... Fs/2)
  - angular $\Omega = 2\pi f$ : **radians/s** (0 ... πFs)
- (so-called) **digital** frequencies
  - $f/Fs$ : **cycles/sample** (0 ... 1/2)
  - angular: $\omega = 2\pi f/Fs$: **radians/sample** (0...π)

# Complex Numbers

- To calculate $a_k$ and $\varphi_k$ efficiently, we need complex numbers
- Complex numbers: $\mathbb{C}$
  - real and **imaginary part** (***i***, *sometimes j*)
  - $i$ is root of $-1$: $\boldsymbol{i^2 = -1}$
  - $c = p + qi, c \in \mathbb{C}, p, q \in \mathbb{R}$
- Euler formula

$$e^{a+i\phi} = e^a(\cos\phi + i\sin\phi)$$

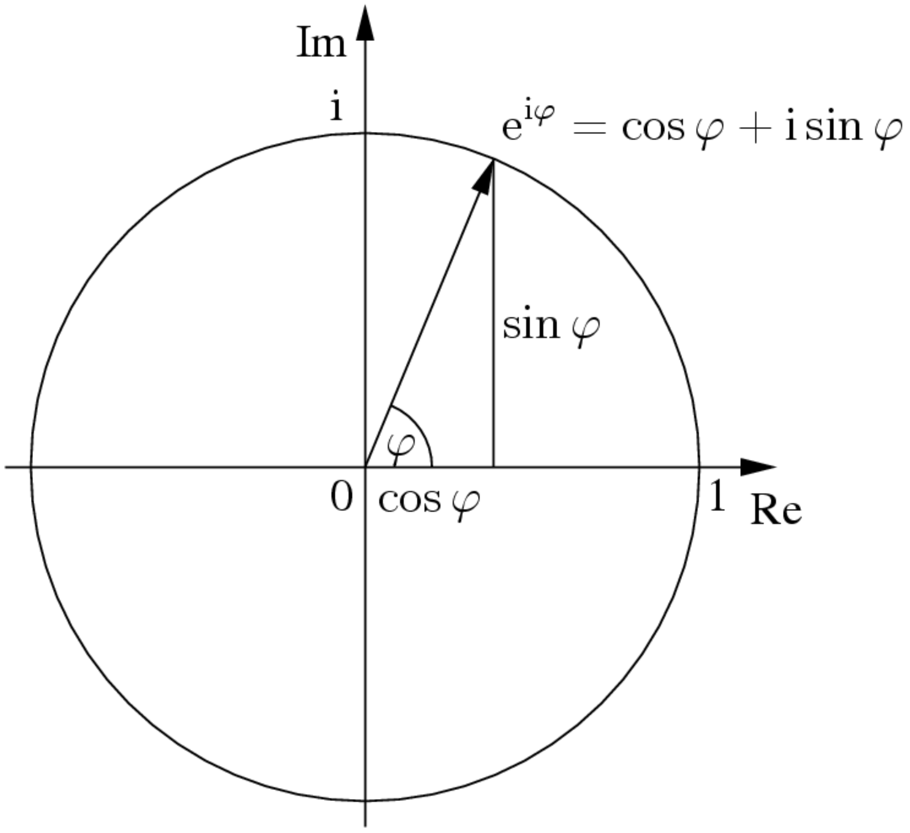
# Trigonometric Functions and Complex Exponentials

*Any complex number c can be represented as* ***magnitude a*** *and* ***angle ϕ***

$$c = p + qi = a\, e^{i\phi}$$

$$\Re\left(e^{i\phi}\right) = \cos\phi$$

$$\Im\left(e^{i\phi}\right) = \sin\phi$$



# Complex Fourier Series

- Again, **x** is a **signal** of length N

$$x[n]=\sum_{k=0}^{N-1} c_k \; e^{i\frac{2\pi}{N}nk}$$

$$a_0=c_0 \;\;,\;\; a_k=2|c_k|,\;\; \phi_k=angle\, c_k$$

- $e^{i\frac{2\pi}{N}nk}$ is complex sine(cosine) with freq *k*
- The **magnitude *a*** and **phase *Φ*** are now **combined** in the **complex coefficients $\mathbf{c_k}$**
- The set of $\mathbf{c_k}$ is called the **spectrum** of *x*

# Discrete Fourier Transform

- By **correlation** with the **complex** sines we get the $c_k$ (up to a normalisation factor) as $X[k]$

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-i\frac{2\pi}{N}nk}$$

- ***X*** is called the (**complex**) **spectrum of** signal ***x***
- The **inverse transform** applies the Fourier series:

$$x[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] e^{+i\frac{2\pi}{N}nk}$$

# Fourier Transform for the Mathematically Inclined

- We can **generalise** to continuous periodic functions (with period T): $$c_k = \frac{1}{T} \int_{\frac{-T}{2}}^{\frac{T}{2}} x(t) e^{-ik2\pi ft} dt$$
- And even do this for **non-periodic** functions by increasing T to infinity: $$X(f) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft} dt$$ $$X : \mathbb{R} \to \mathbb{C}$$
- ***X*** is called the **continuous Fourier Transform** of **x**
  ***X*** contains positive and **negative frequencies**

# Inverse Fourier Transform (still for the math buffs)

- **Inverting** the **Fourier Transform** leads to a **generalised Fourier Series**:

*Fourier Series*

$$x(t) = \sum_{k=-\infty}^{\infty} c_k e^{ik2\pi ft}$$

*Inverse Fourier Transform*

$$x(t) = \int_{-\infty}^{\infty} X(f) e^{i2\pi ft} dt$$

- **X(f)** gives the **magnitude** (absolute) and **phase** (angle) for every frequency in the continuum and is called the **(continuous) spectrum** of **x(t).**

# Spectra of Digital Signals

- The **frequencies k** in the spectrum are **relative** to the **length *L*** of the signal,

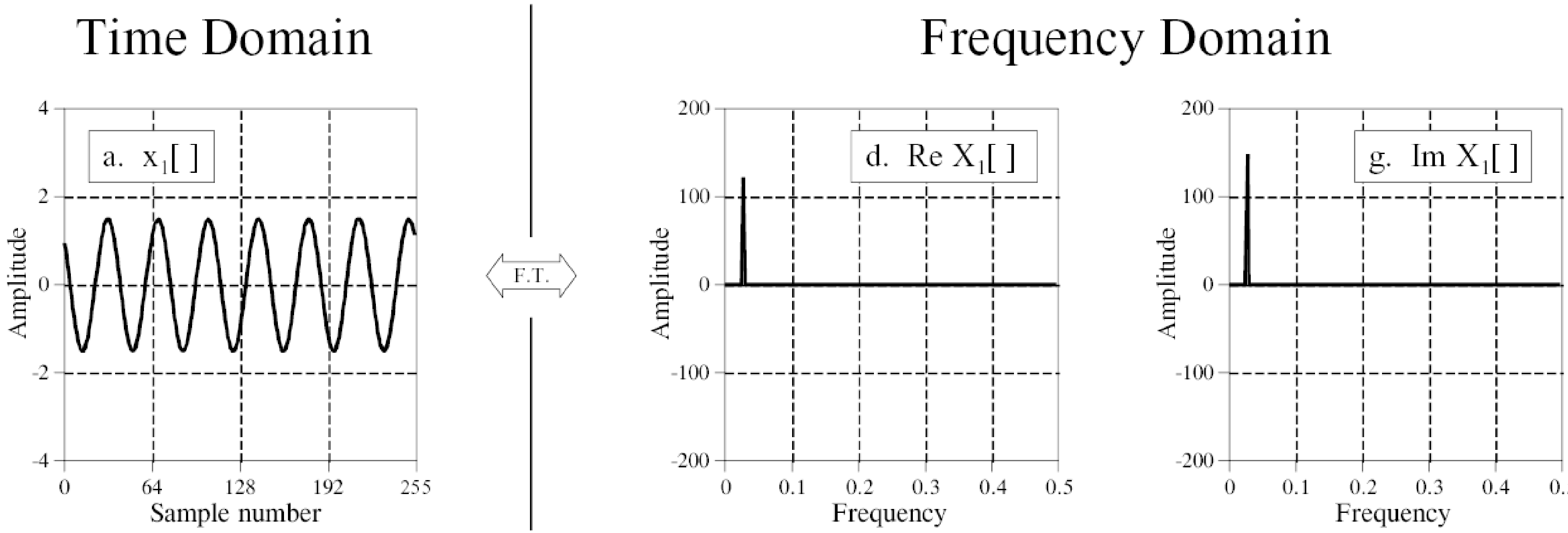
  i.e. ***X*[*k*]** in the spectrum corresponds to **k cycles over length *L*** of the whole signal.
- Dividing ***k*** by length ***L*** gives the **digital** frequency (**cycles per sample**) ***fd*** = ***k/L***
- **Multiplying** by the **sampling frequency *Fs*** gives the frequency ***f*** in **Hertz**: ***f*** = ***Fs*** **·** ***k*** / ***L***

# Real and Complex FT

- **Complex FT** elegant maths
  - **spectra** with positive/**negative frequencies**
  - **symmetric** around the **y-axis** (more later)
- **Real FT**
  - describes the **whole process** in **real maths**
  - a bit more messy, but less complex ;-)

# Example Spectra

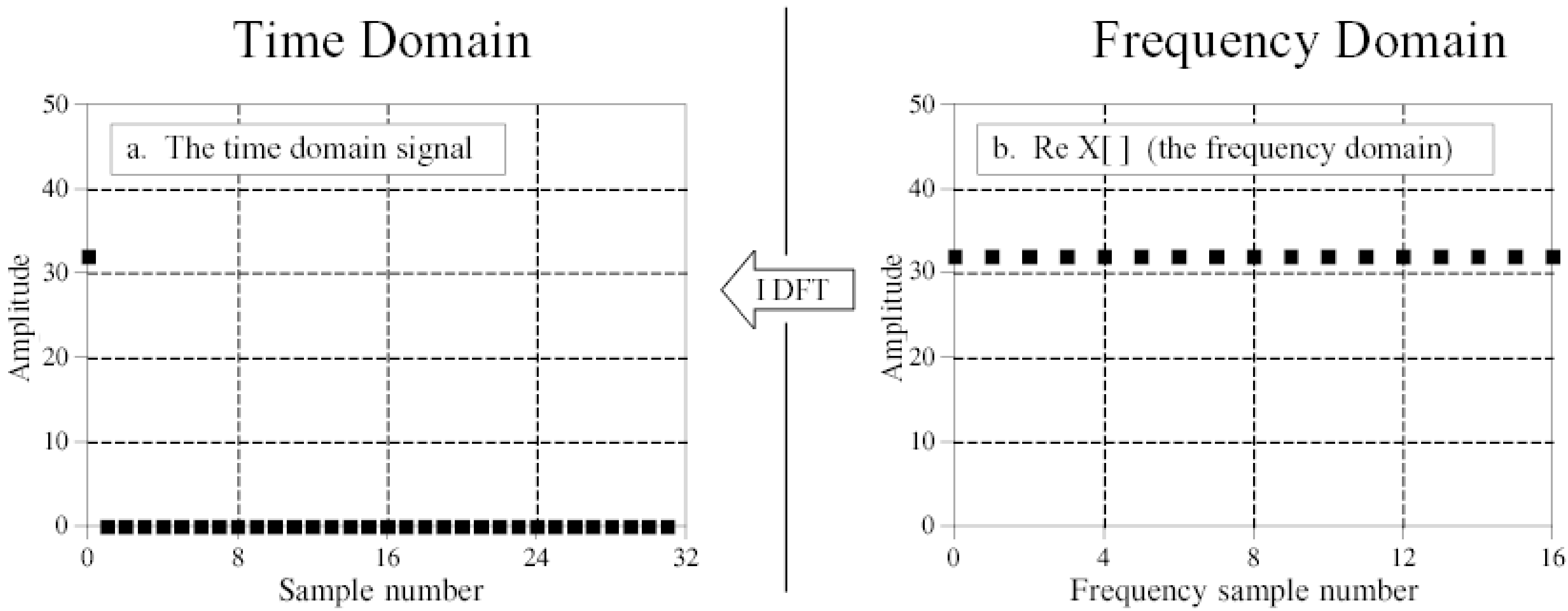
- Discrete Fourier Transform of a **single sinusoid**:




- **Phase = balance of Re and Im**, depends on time-axis **shift** of x

# Spectra 2

- Fourier Transform of a **single impulse**:




- One **impulse** contains **all frequencies**

# Notation of the FT

- The **Fourier Transform** of a **signal** (function) is denoted by $(\mathcal{F}x)(f)$, often written as $\mathcal{F}(f)$, if clear in the context
- There is a **special symbol** $\multimap$ used for the FT, like this: $x(t) \multimap X(f)$
  or in short $x \multimap X$

# Linearity of the FT

- The Fourier Transform is **linear** (**invariant** under **addition** and **scalar multiplication**):

$$F(x_1+c\,x_2)(f)=F(x_1)+c\,F(x_2)(f)$$

- Using the transformation symbol, linearity can be written like this:

$$x_1+c\,x_2 \circ\!\!-\!\!\bullet\, X_1+c\,X_2$$

# Parseval's Theorem

- **Parseval**'s theorem: the **spectrum** has the **same energy** as the **signal**:

$$\sum_t \left|\left(x(t)\right)\right|^2 = \frac{1}{N} \sum_f \left|\left(\left(\mathcal{F}x\right)(f)\right)\right|^2$$

- or using the transformation symbol:

$$\sum_t \left|\left(x(t)\right)\right|^2 = \frac{1}{N} \sum_f \left|\left(X(f)\right)\right|^2, \text{ where } x \circ\!\!-\!\!\bullet X$$

- The **division by *N*** is because we **normalise only** the **inverse FT** (see definition)

# Symmetries of the FT

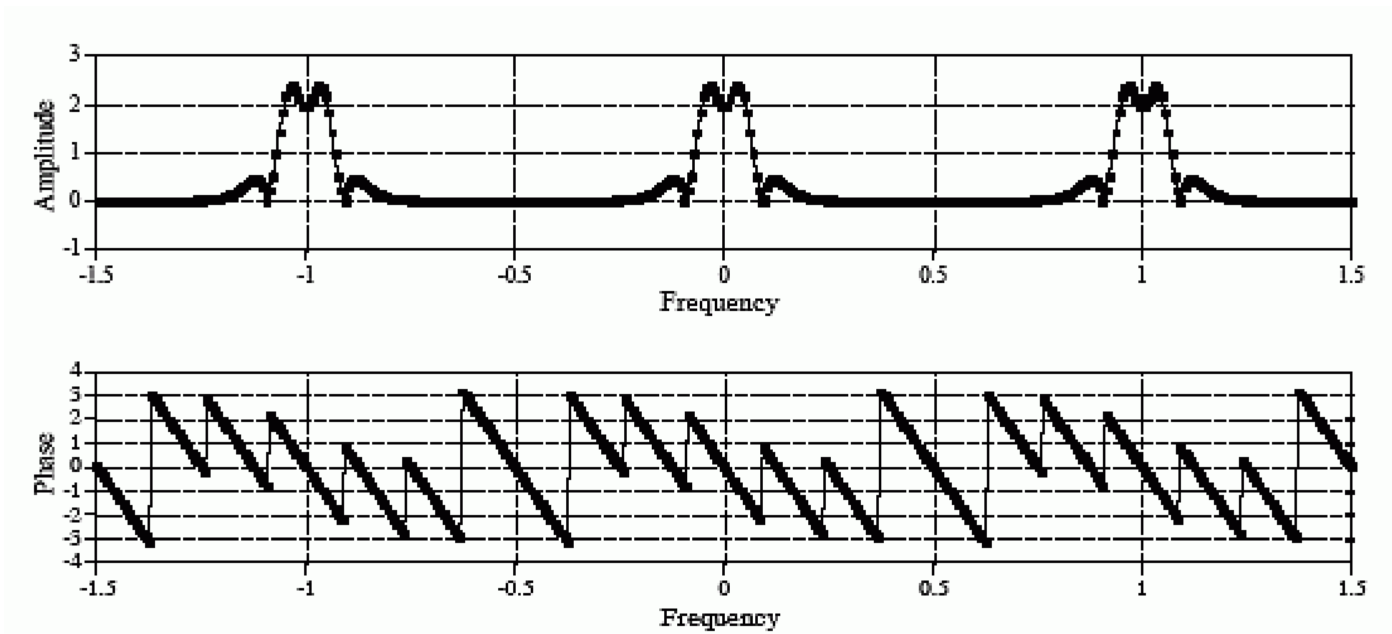
- Spectra of **real** signals (e.g. all measured signals) have several **symmetry** properties:

$$x(t): real \text{ then } \left\{ \begin{matrix} |(X(f))|: even & arg(X(f)): odd \\ \Re(X(f)): even & \Im(X(f)): odd \end{matrix} \right\}$$

**Even** functions are **mirror**-symmetrical around the y-axis, **odd** functions are **point**-symmetrical around the origin.

- For **real** signals, **half** of the **spectrum** contains **all information** because of these symmetries.

# Symmetries of the FT




- The **magnitude** (even function) is mirror-symmetrical
- The **phase** (odd function) is point-symmetrical

# Time and Frequency

- **Stretching** the signal over **time** (*c*>1) **compresses** the **spectrum** over **frequency** and reduces **magnitude**

$$x(ct) \circ\!\!-\!\!\bullet \frac{1}{|c|} X\left(\frac{f}{c}\right)$$

- **Shifting** a signal over **time** leaves the **frequencies unchanged** but **modifies phase**

$$x(t-t_0) \circ\!\!-\!\!\bullet X(f)\, e^{i2\pi f t_0}$$

# Fast Fourier Transform

$$X[k] = \sum_{n=0}^{N-1} x[n] e^{-i\frac{2\pi}{N}(n)k}$$

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} x[2n] e^{-i\frac{2\pi}{N}(2n)k} + \sum_{n=0}^{\frac{N}{2}-1} x[2n+1] e^{-i\frac{2\pi}{N}(2n+1)k}$$

$$X[k] = \sum_{n=0}^{\frac{N}{2}-1} x[2n] e^{-i\frac{2\pi}{N}(2n)k} + e^{-i\frac{2\pi}{N}k} \sum_{n=0}^{\frac{N}{2}-1} x[2n+1] e^{-i\frac{2\pi}{N}(2n)k}$$

$$X\left[k+\frac{N}{2}\right] = \sum_{n=0}^{\frac{N}{2}-1} x[2n] e^{-i\frac{2\pi}{N}(2n)k} - e^{-i\frac{2\pi}{N}k} \sum_{n=0}^{\frac{N}{2}-1} x[2n+1] e^{-i\frac{2\pi}{N}(2n)k}$$

FFT of even indices **'twiddle factor'** FFT of odd indices

X[k] and X[k+N/2] **differ only** by the **sign** of the **'twiddle factor'**. By recursively reducing the problem to halves, we can perform a complete FT in **O(n log n)**

# Take-Home Messages

- **Dot product** (**raw correlation**) - a form of signal similarity
- ***Fourier Transform***
  - **decomposes** a **signal** using its **correlation** with **sinusoids**
  - sinusoid **frequencies** are **integer multiples** of **1/len(signal)**
  - produces a **spectrum** with as many points as the signal
  - always has an **inverse**
  - is **linear**
- Points on the **spectrum** are **complex numbers** with
  - **magnitude** (~amplitude), and
  - **phase** (~angle, time-shift, balance of sine/cosine)
- **F**ast **F**ourier **T**ransform (fft,ifft) is **efficient O(n log n)**

# READING

Smith, S.: The Scientist and Engineer's Guide to DSP. Chapter 8. http://www.dspguide.com/ch8.htm