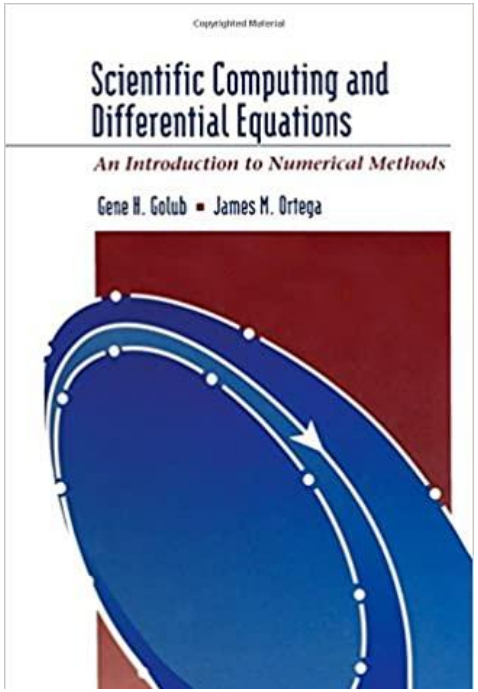
UNIVERSITY OF PORTSMOUTH

Introduction to Numerical Fourier Analysis

HAMIDREZA KHALEGHZADEH

What is Scientific Computing?

The question is answered in *Scientific Computing and Differential Equations – An Introduction to Numerical Methods,* an excellent book emphasizing the importance of complex scientific applications, such as differential equations, on a computer:



Scientific Computing is the collection of tools, techniques, and theories required to solve mathematical models of problems in Science and Engineering on a computer

Most of these tools, techniques, and theories originally developed in Mathematics, and is called Numerical Analysis (or Numerical Mathematics).

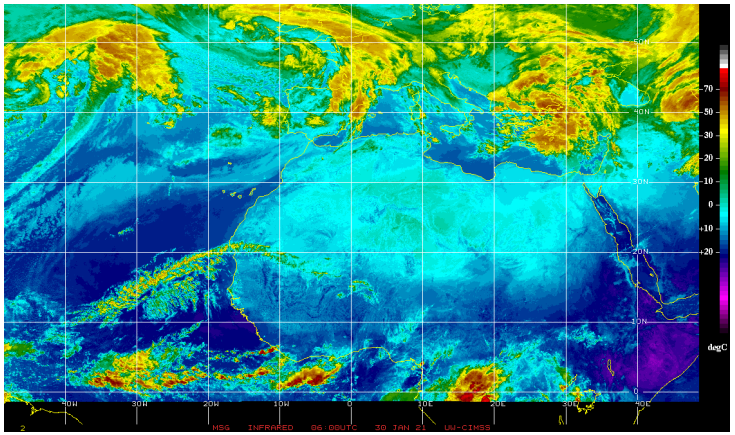
Scientific Computing and Simulation is the multi-disciplinary field of computer-based modelling and simulation for studying scientific phenomena and engineering designs

Why Scientific Computing?

There are two main approaches to gain scientific knowledge:

1. Theoretical investigation
   - Developing hypothesis and models
   - Analytical calculations
     - Models are usually too complicated or even impossible to solve
     - Only applicable to simple scenarios
2. Experimentation
   - Building models
   - Predicting theoretical results
   - Comparing experimental outcomes with the predicted results
     - Might be impossible to do
     - Might be dangerous or unwelcome
     - Might be very expensive (in time/money)

Applications



Meteorology

- Weather forecasts
- Simulation of hurricanes and storm surges

Modelling climate change

- Greenhouse effect
- Ocean currents (gulf stream, etc.)

Image from http://tropic.ssec.wisc.edu/

Applications

Invistigate the stability of buildings

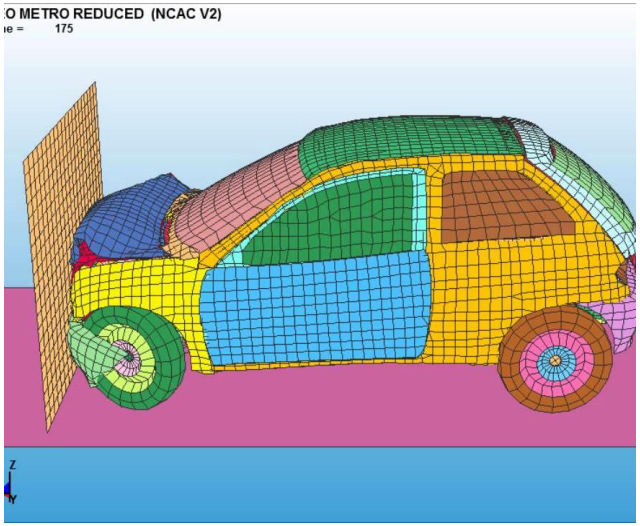
How buildings can resist

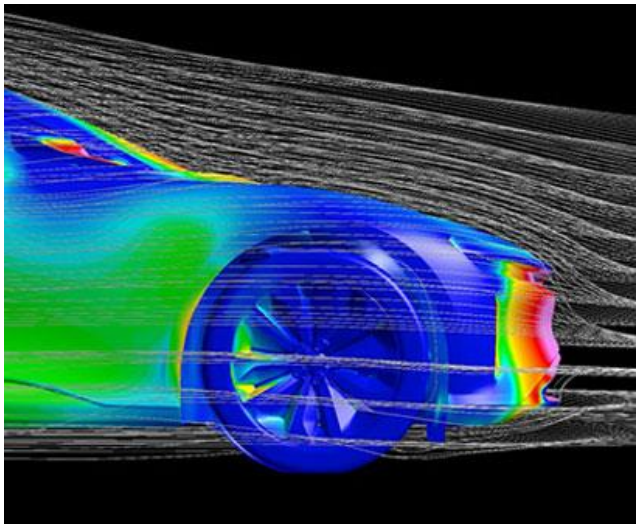
- wind loads
- earthquakes

Applications

Aviation and Astronautics

- flight path of space crafts or satellites
- re-entry of space crafts





Applications

Car industry

- Aerodynamics
- Crash tests
- Assembly of parts
- Build prototypes or rather simulate?

Scientific Computing and Simulation Module

Syllabus will include:

- Part I: Fast Fourier Transform and its applications in image processing
- Part II: Formulate and solve physical problems using numerical methods and simulation models

Assessment

Coursework consisting of a two-part portfolio:

- 65% the record of activities carried out in the weekly lab scripts (“lab book”).
- 35% the results of a mini-project on scientific computing and simulation undertaken in the second part of the term.
- 4000 words
  - **Scientific Computing and Simulation Coursework**

Introduction to Numerical Fourier Analysis

Goals

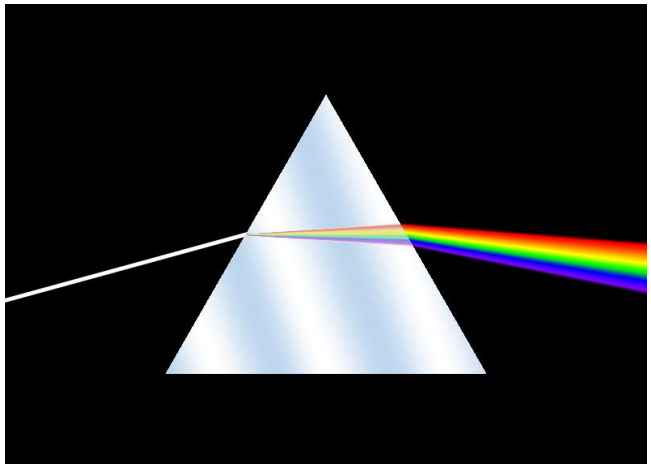
Introduce the basic concepts and equations of Fourier analysis, to be used later in interesting kinds of image reconstruction.

Decomposing Waveforms

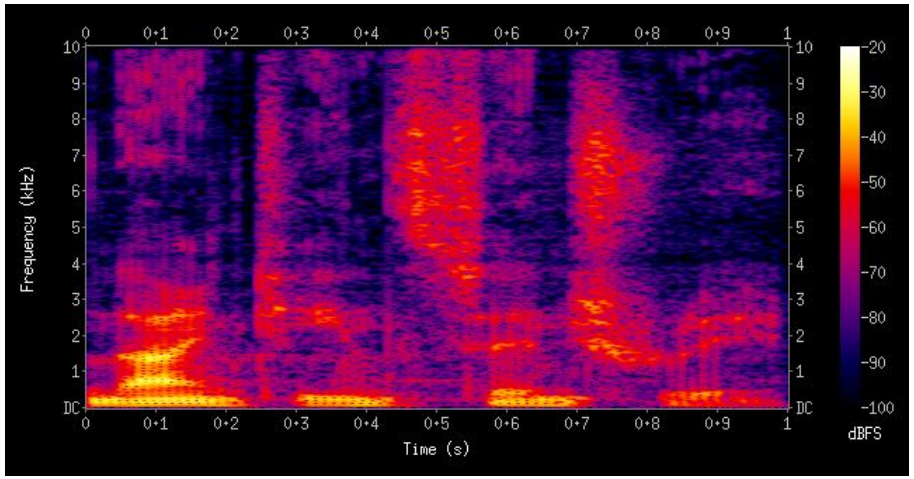
General physical idea of dividing waves like sound or light into different frequency/wavelength components presumably familiar.

Examples:

Spectroscopy (light)



Sound spectrogram (e.g. voice print)



Decomposing Waveforms

Decomposing a combinational wave $x[n]$ into a group of additive components $X_1[n]$, $X_2[n]$, and $X_3[n]$.

where:

$$\mathrm{X}[n] = X_1[n] + X_2[n] + X_3[n]$$

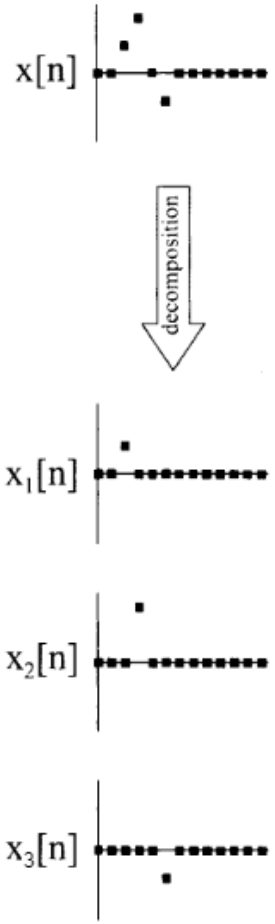


Figure form Smith S. Digital signal processing: a practical guide for engineers and scientists. Elsevier; 2013 Oct 22.

Why Decomposition?

Replacing a complicated analysis task with several easier ones

- Component waves are simpler than the original wave
- Ex: $2041 * 4 = (2000 + 40 + 1) * 4$

Removing undesirable components form original waves

- Ex: Noise elimination

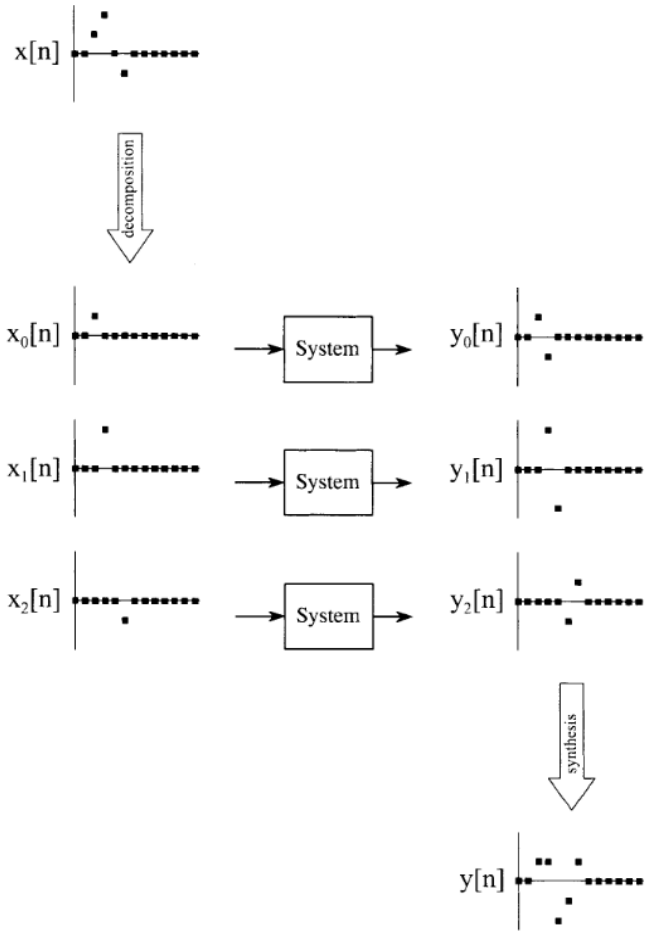


Figure form Smith S. Digital signal processing: a practical guide for engineers and scientists. Elsevier; 2013 Oct 22.

How Decompose

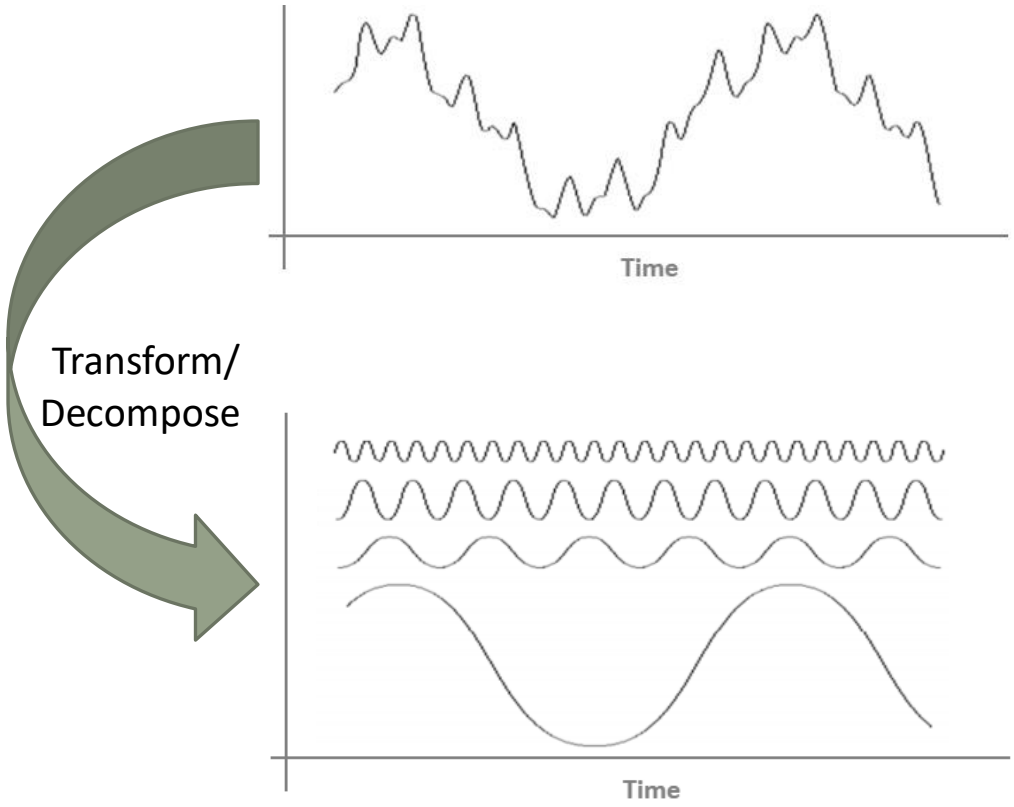
The goal of waveform decomposition is to replace a complicated problem with several easy ones.

There are two main ways to decompose

1. Fourier decomposition / Fourier Transform
2. Impulse decomposition

Fourier Transform

Fourier Transform is one of the most powerful tools which enables us to find the spectrum of a wave.

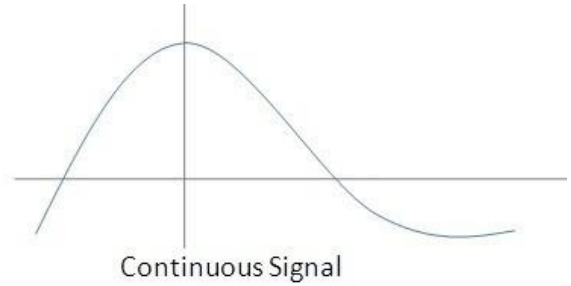


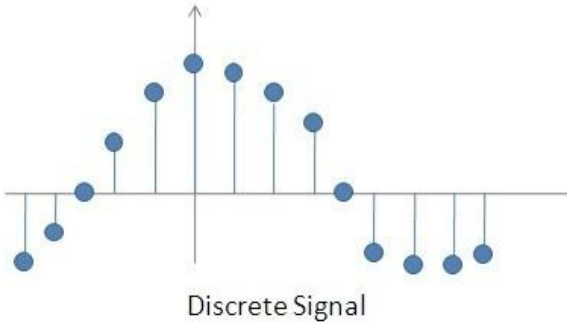
Watch a Video

But what is the Fourier Transform? A visual introduction.

Recap

A signal can be either continuous or discrete





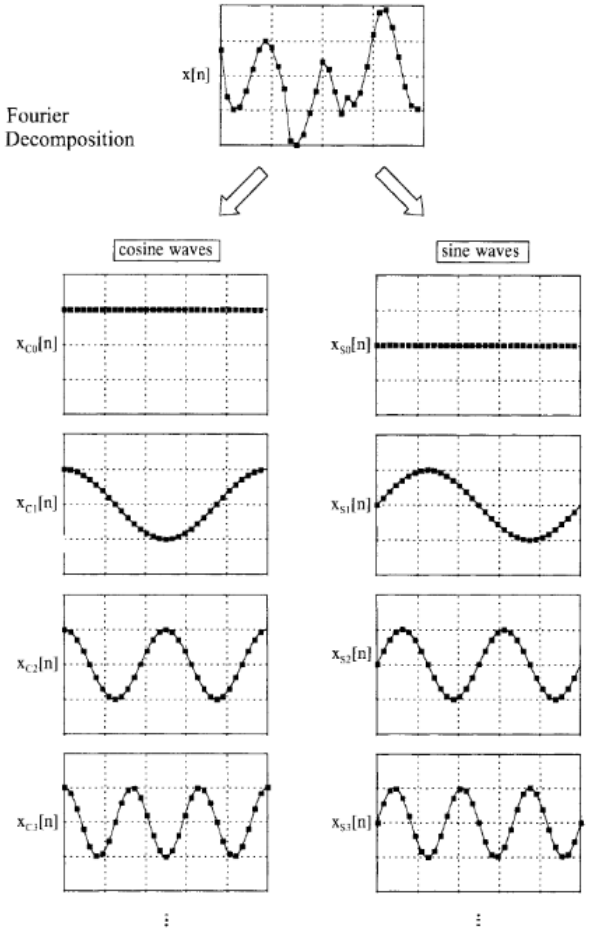
Discrete Fourier Transform (DFT)

Fourier Transform breaks general (but periodic) waveforms into a series of pure waves of fixed frequency.

- Half of them sine waves
- Half of them cosine waves

Why discrete Fourier Transform?

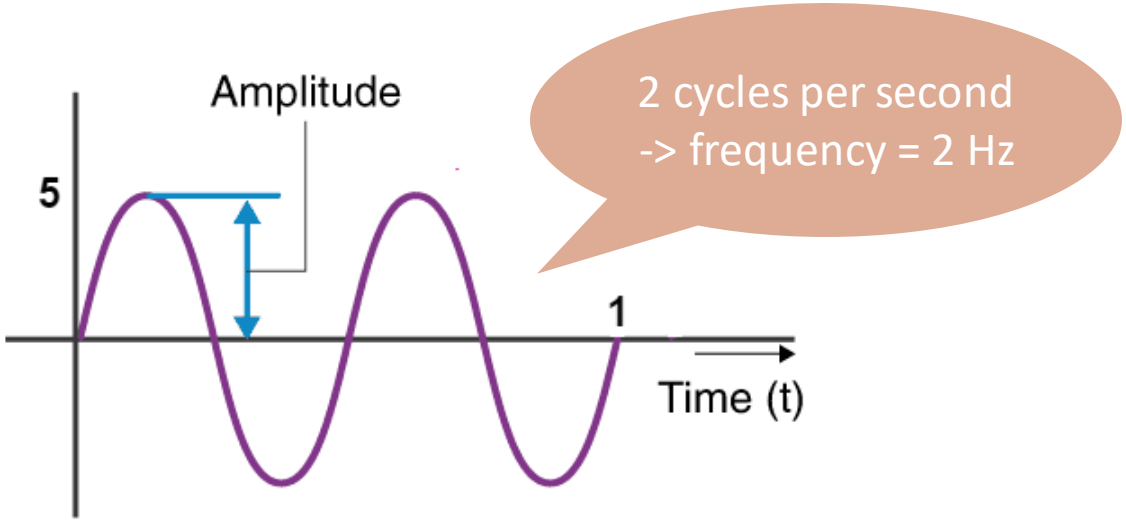
- Computers can only work with information that is discrete and finite in length.



Sinusoidal Wave Features

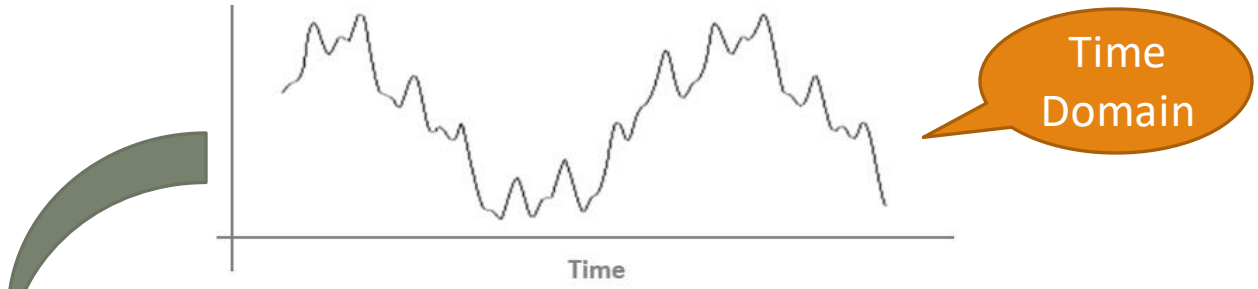
Each sinusoidal wave has two main parameters:

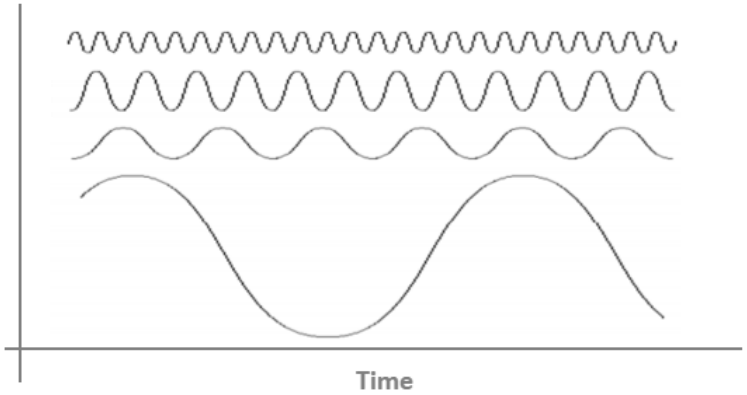
- Frequency (f)
  - The number of oscillations per unit time
- Amplitude
  - The maximum distance moved by a point on a wave measured from its equilibrium position

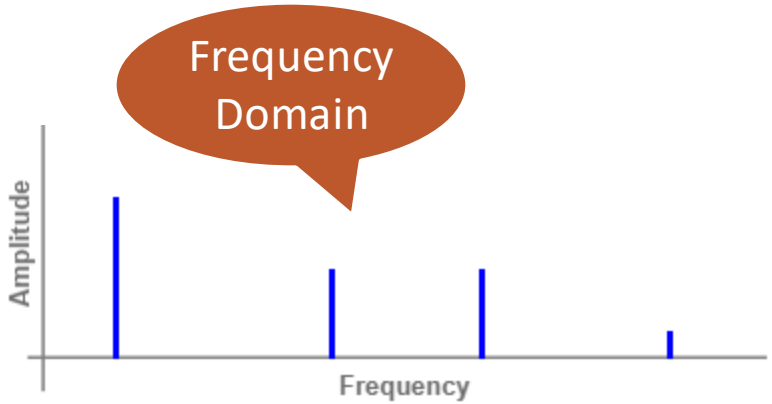


Time and Frequency Domains

Fourier Transform convert Time Domain to Frequency Domain.







Time to Watch

How the Fourier Transform Works

DFT Series

Consider an input signal X consisting of N points / samples. DFT decomposes it into cosine waves (components) and sine waves.

- Each wave has a *frequency* and *amplitude*.

$$X_t = \sum_{k=0}^{N-1} A_k \cos(\frac{2\pi kt}{N}) + B_k \sin(\frac{2\pi kt}{N})$$

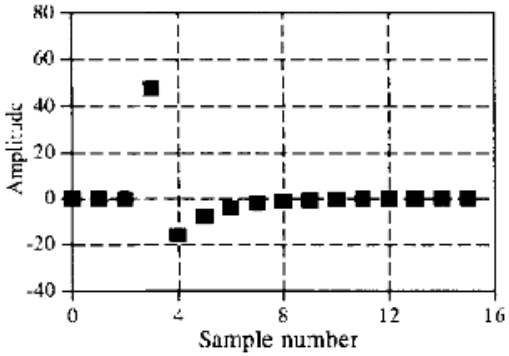
A_k holds the amplitude of the k-th cosine wave

- Real part of wave component wave

B_k holds the amplitude of the k-th sine wave

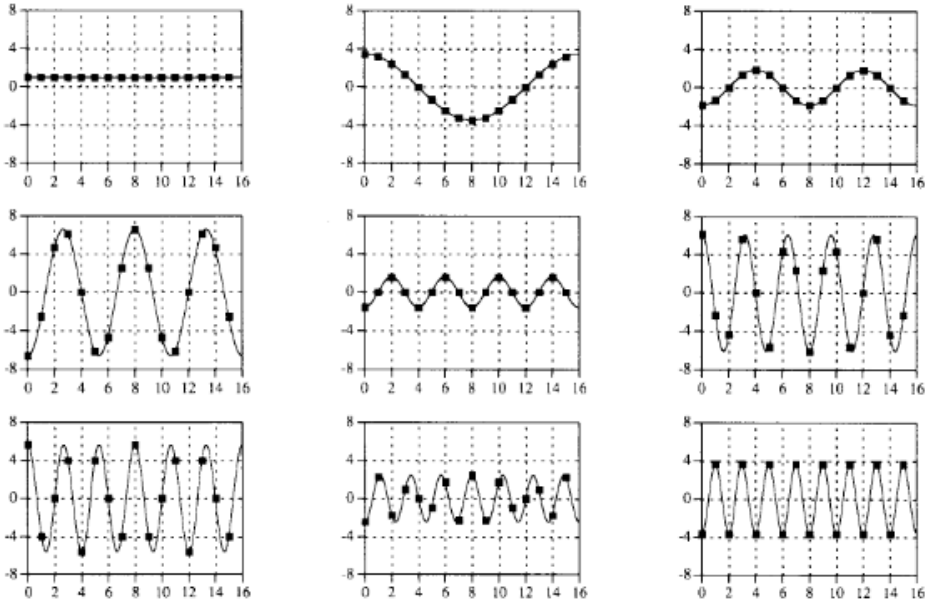
- Imaginary part of the component wave

Frequency of each wave is $\frac{k}{N}$

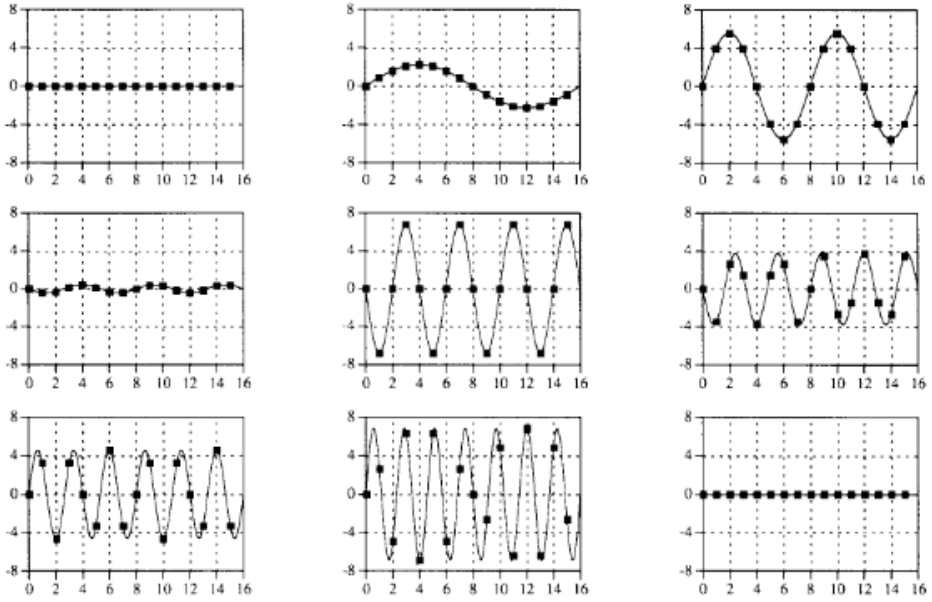


DECOMPOSE

Cosine Waves



Sine Waves



Complex DFT

Fourier transforms can be subdivided into real and complex versions.

Real DFT is the simplest, using ordinary numbers and algebra for the synthesis and decomposition.

- Discussed in the previous slides

Complex DFT is more general version of the real DFT and represents data using complex numbers.

Complex Numbers

Recall complex numbers have the general form:

$$z = a + bi$$

where *a* is the "real part", *b* is the "imaginary part", and *i* is the square root of -1.

The "complex conjugate" of this number is:

$$z^* = a - bi$$

For addition of complex numbers, just add real and imaginary parts separately:

$$(a + ib) + (c + id) = (a + c) + i(b + d)$$

Multiplication goes like this:

$$(a + ib)(c + id) = (ac - bd) + i(ad + bc)$$

Euler's Relation

Euler's relation establishes the relationship between the sine and cosine / trigonometric functions and the complex polar functions.

$$e^{ix} = \cos(x) + i\ \sin(x)$$

$$e^{-ix} = \cos(x) - i\ \sin(x)$$

Consequently:

$$\cos(x) = \frac{e^{ix}+e^{-ix}}{2} \qquad \text{and} \qquad \sin(x) = \frac{e^{ix}-e^{-ix}}{2i}$$

This formulate let's us express the DFT in terms a single set of complex amplitudes.

Complex DFT

Consider an input discrete signal X consisting of N complex points. The complex DFT Series can now be written as:

$$C_k = \frac{1}{N}\sum_{n=0}^{N-1} X_n e^{-2\pi kni/N} \quad , k = \{0, 1, \dots, N-1\}$$

The two amplitude sets of real numbers A_k and B_k have been replaced by a single set of complex numbers, C_k.

We will refer to the parameter k of the DFT as the "wave number"

Transforming the original signal into its components is called (forward) Fourier transform.

- It gives Fourier components of the input signal.

Rectangular Complex DFT

Rewriting the polar complex DFT in rectangular format using Euler's relation:

$$C_k = \frac{1}{N} \sum_{n=0}^{N-1} X_n \left(\cos(2\pi kn/N) - i \ \sin(2\pi kn/N)\right)$$

Inverse Complex DFT

If we know the Fourier components (C_k) of a given function X, consisting of N components, and want to reconstruct it, we can use the formula:

$$X_n = \sum_{k=0}^{N-1} C_k e^{2\pi kni/N} \ , n = \{0, 1, \dots, N-1\}$$

Reconstructing an original signal from its components is called inverse Fourier transform.

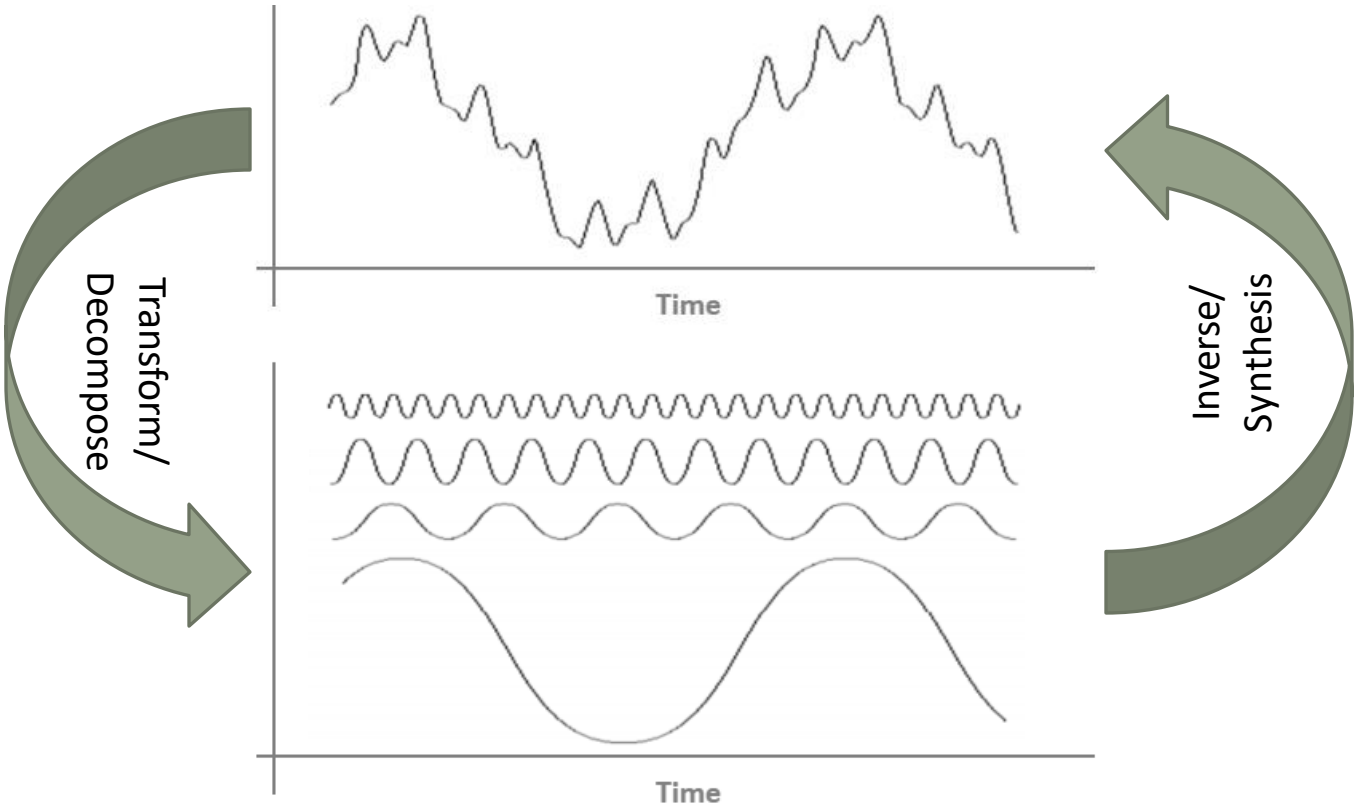
- Reconstructing the signal from its Fourier components.

Rectangular Inverse Complex DFT

Rewriting inverse complex DFT in Rectangular form using Euler's relation:

$$X_n = \sum_{k=0}^{N-1} C_k \left(\cos\left(2\pi kn/N\right) + i \ \sin(2\pi kn/N)\right)$$

Transpose vs Inverse



Spatial Transforms

1D DFT has been discussed by now

- Ex: Voice processing

Fourier transform can be generalized to higher dimensions

2D DFT

- Ex: Image processing

2D DFT

Consider a given 2D signal, stored in an N*N matrix X.

Forward 2D DFT:

$$C_{kl} = 1/N^2 \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} X_{mn} \cdot e^{-2\pi i(km+nl)/N}$$

C is a 2D matrix where C_{kl} stores the Fourier component for the wave number k and l.

Inverse 2D DFT:

$$X_{mn} = \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} C_{kl} \cdot e^{2\pi i(km+nl)/N}$$

Image Fourier Transformation

Matrix X stores the image

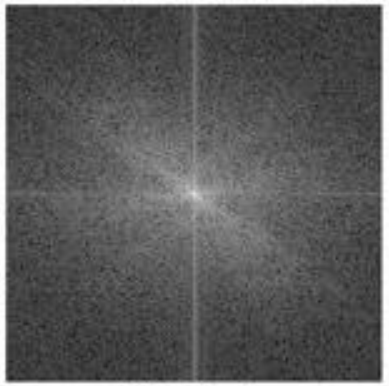
- X[m][n] is correspond to pixel values of the image

Matrix C stores the result Fourier components

- C[k][l] is the Fourier components (spatial frequency components) associated to wave number k and l.

Image X

2D DFT



Matrix C

So What?

During the first part of this module, we will use Fourier transforms to do simple filtering of images and interesting kinds of medical and scientific image reconstruction.

Recap

1D DFT:

$$C_k = \frac{1}{N}\sum_{n=0}^{N-1} X_n e^{-2\pi kni/N} \quad , k = \{0, 1, \dots, N-1\}$$

$$C_k = \frac{1}{N}\sum_{n=0}^{N-1} X_n \left(\cos(2\pi kn/N) - i\ \sin(2\pi kn/N)\right)$$

1D Inverse DFT:

$$X_n = \sum_{k=0}^{N-1} C_k e^{2\pi kni/N} \quad , n = \{0, 1, \dots, N-1\}$$

$$X_n = \sum_{k=0}^{N-1} C_k \left(\cos\left(2\pi kn/N\right) + i\ \sin(2\pi kn/N)\right)$$

Further Reading

Smith S. *Digital signal processing: a practical guide for engineers and scientists*. Elsevier; 2013 Oct 22.