# Mobile Sensing: Sensing, Sampling, and Filtering

**Partly based on: "CS390MB:Mobile Health Sensing and Monitoring" by Deepak Ganesan, UMASS**

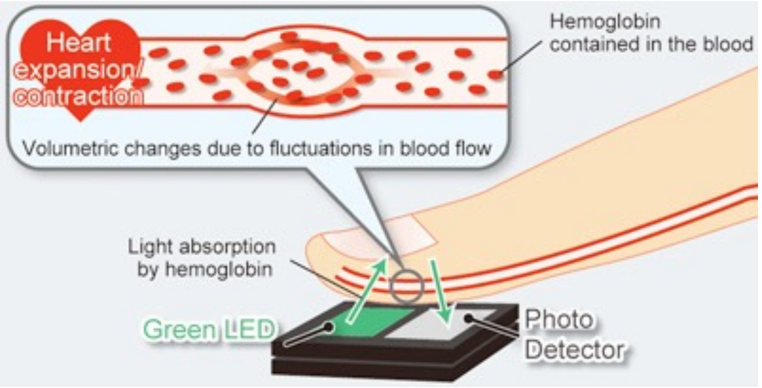
Master studies, Winter 2021/2022


Dr Veljko Pejović
Veljko.Pejovic@fri.uni-lj.si

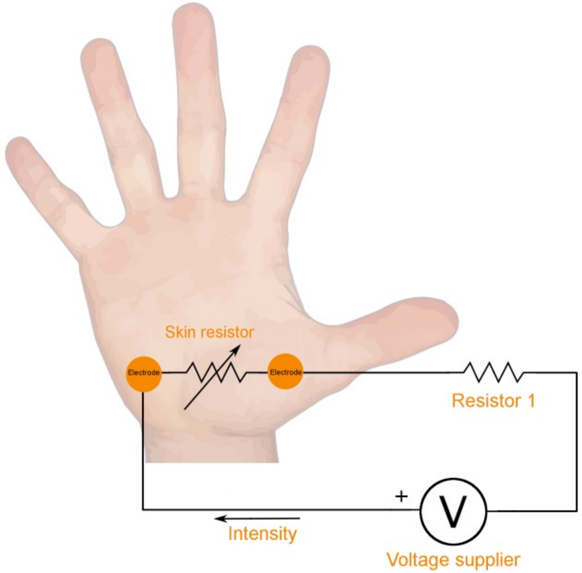
University of Ljubljana
Faculty of Computer and Information Science

# Sensing

- Sensing – measures a physical quantity
  (Actuating – changes a physical quantity)
- A sensor often produces voltage proportional to the physical quantity being measured
- Different underlying mechanisms




Photoplethysmogram (PPG) from rohm.com




Galvanic skin response from tobiipro.com

University of Ljubljana
Faculty of Computer and Information Science

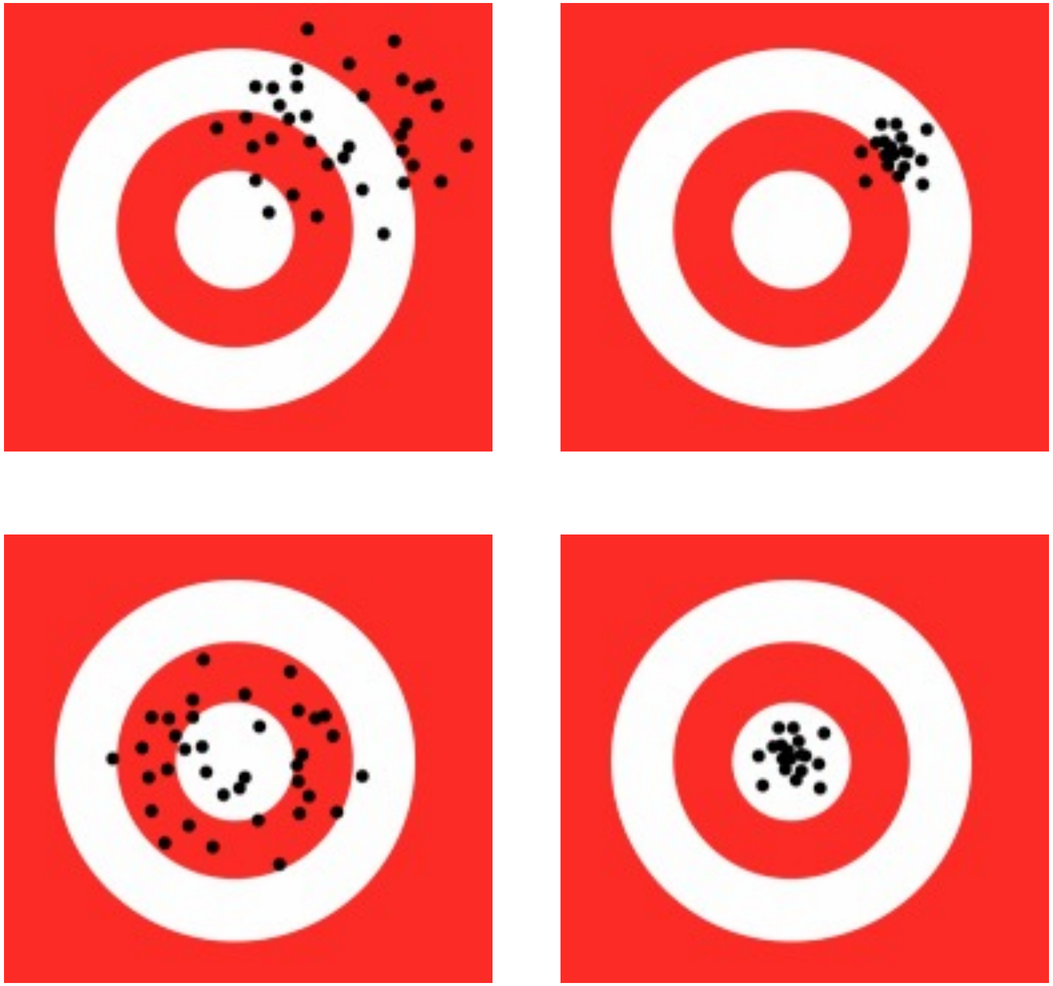
# Sensor Properties

- Sensing range
  - Measurements are valid only within a limited range between L and H

$$f(x(t)) = \begin{cases} ax(t)+b & L \le x(t) \le H \\ aH+b & x(t) > H \\ aL+b & x(t) < L \end{cases}$$

- Accuracy
  - Max difference between the actual measured value and what the sensor outputs (see a and b above)
- Precision
  - Degree of reproducibility
    - If the same input value is measured a number of times, an ideal sensor would always output the same output value

# Sensor Properties



Source: https://www.instrumentation.co.uk

# Sensor Properties

- Resolution
  - The number of discrete values that can be represented over the range of analog values
- Sensitivity (s)
  - The smallest absolute difference between two values of a physical quantity whose sensor readings are distinguishable

# Sensor Properties

- Dynamic range (D)

$$D = \frac{H - L}{s}$$

Usually in decibels:

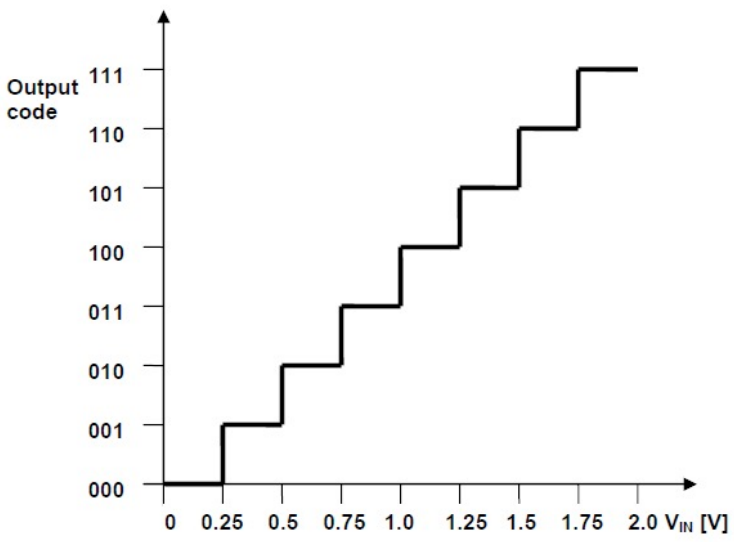
$$D_{dB} = 20\log\left(\frac{H - L}{s}\right)$$

What's "Decibel"?

- When assembling your own sensing system:
  – Sensor data sheets should list most of these metrics
- When using a commodity phone/wearable:
  – Often very little information, low precision/accuracy

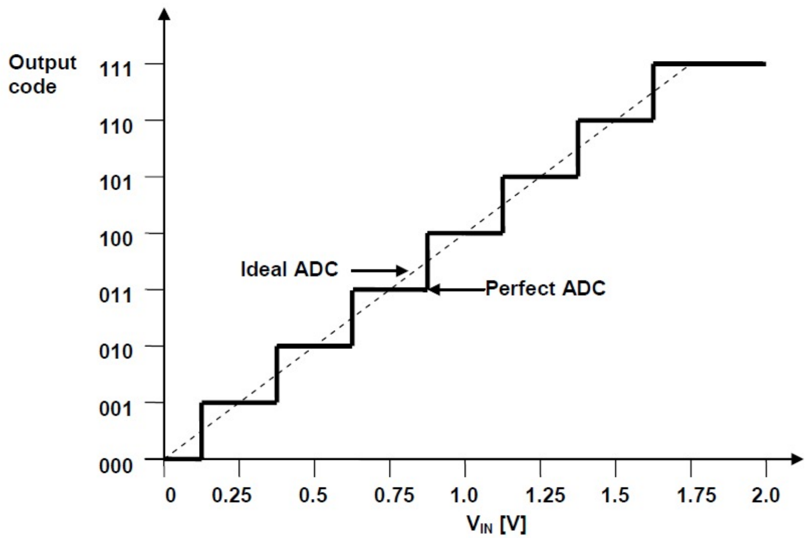
# From Analog to Digital Domain

- Modern sensors are either inherently digital (i.e. output numbers) or have their analog values converted to digital via the computing device's analog-digital converter (ADC)



# From Analog to Digital Domain

- Quantization error:
  - An n-bit sensor can output only $2^n$ values
  - Quantization limits precision
    - Calculate the root mean square quantization error
  - What is the effect on the dynamic range?
    - Each additional bit yields 6 dB of dynamic range



# From Analog to Digital Domain

- Sampling
  - Take readings of the signal in certain moments only
- Sampling rate
  - The rate at which the readings are taken
- Signal frequency
  - How fast does the signal change
- How fast do you need to sample in order to capture varying signal?

Limitation of the ADC, the device's CPU, or bus lines

# Nyquist Limit

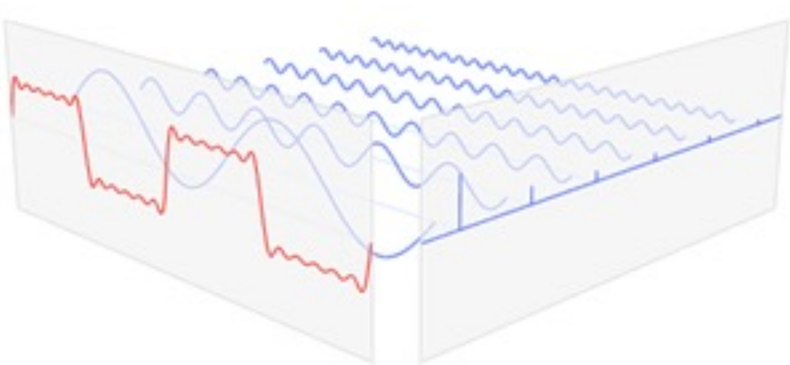
$$f_s \geq 2f$$

$f_s$ – sampling frequency
$f$ – maximum signal frequency

- If you sample below Nyquist limit you cannot reconstruct the signal!
- Aliases
  - Different signals become indistinguishable from one another because the sampling rate was too low to capture the differences

# Time vs Frequency Domain

- Time domain
  - Shows how a signal varies in time
- Frequency domain
  - Shows how much of the signal lies within each given frequency band over a range of frequencies
- Example:

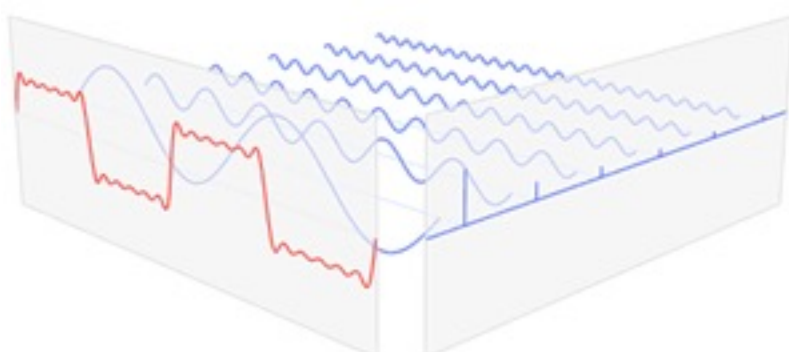


Source: Lucas V. Barbosa, Wikipedia

# Fourier Transform

- Any periodic waveform can be expressed as the sum of an infinite set of sine waves
  - Fourier coefficients: complex numbers that multiply their respective sine waves in order to obtain the original signal (how prominent a particular frequency is in the signal)
  - Use Fast Fourier Transform (FFT) algorithm!

What about non-periodic?



Source: Lucas V. Barbosa, Wikipedia

# Sensor Data Filtering

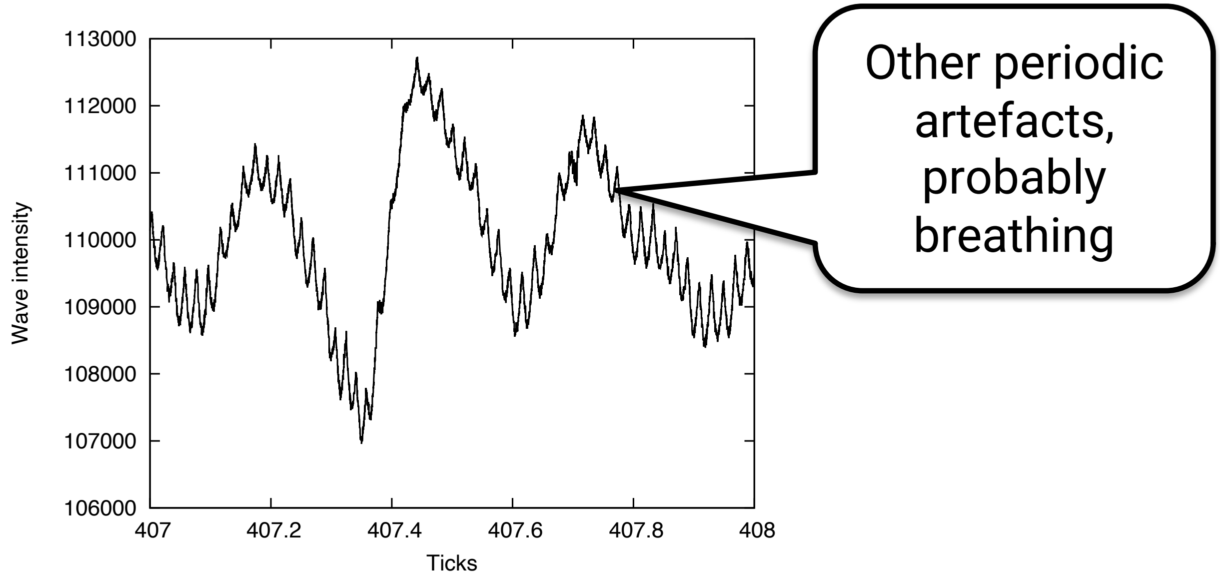
# Noise in Data

- Anything that corrupts the signal and makes it difficult to extract the information of interest
- Sources:
  - Electronic and mechanical noise in a sensor
  - Quantization noise
  - Thermal noise
  - External vibrations
  - Electronic interference
  - Noise due to sensor placement
  - …

University of Ljubljana
Faculty of Computer and Information Science

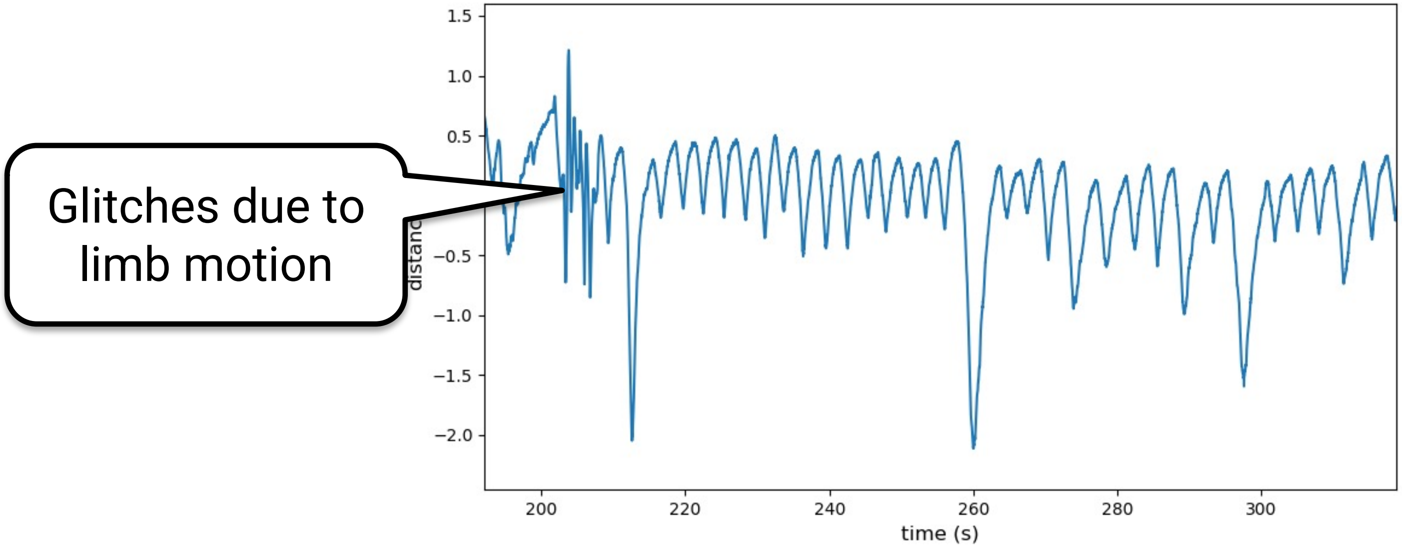
# Noise – Examples

- Heart rate measurements with a wristband



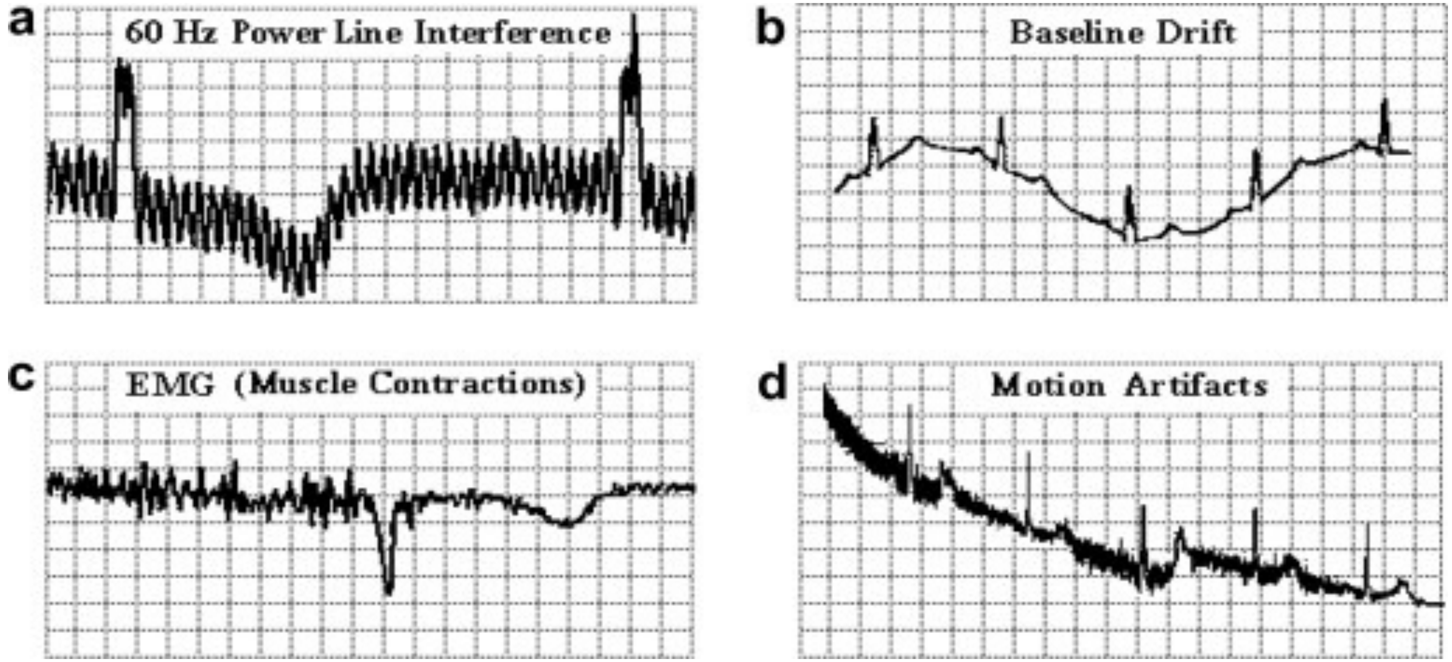
# Noise – Examples

- Breathing measurements via a wireless radar



# Noise – Examples

- Electro cardiogram (ECG) measurements



# Smoothing and Filtering (Time)

TIP: First try to identify the source of noise

- Oversampling and averaging
  - Noise is often random
  - Sample at a rate *N* times higher than what you need and average each *N* readings out
  - Noise should be reduced by a factor of $1/\sqrt{N}$
- Moving average smoothing
  - Average N consecutive values out, and then move the window one step ahead
  - Too large of a window might smooth out the information!

$$s_1 = (x_1 + x_2 + x_3)$$
$$s_2 = (x_2 + x_3 + x_4)$$
$$s_3 = (x_3 + x_4 + x_5)$$
$$\dots$$

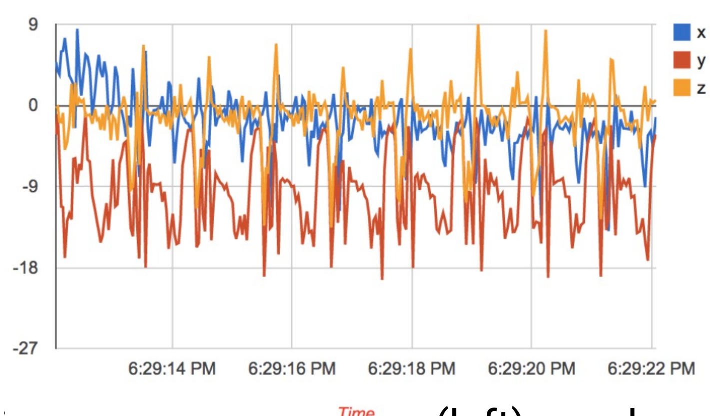
# Smoothing and Filtering (Time)

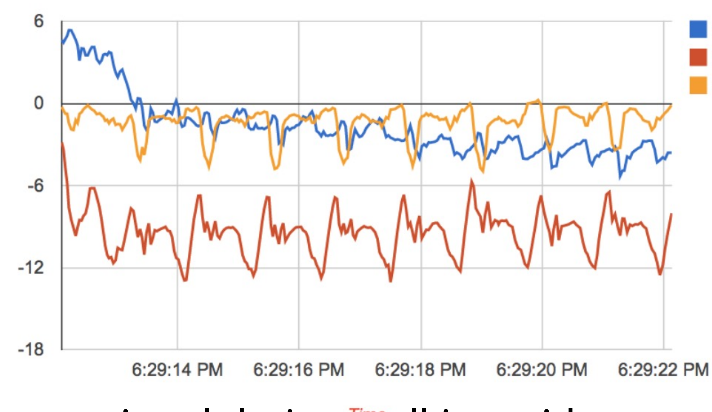
- Exponential smoothing
  - Put more trust in recent samples
    (α – smoothing factor)

    $$s_1 = x_1$$

    $$s_t = \alpha x_t + (1-\alpha)s_{t-1} = s_{t-1} + \alpha(x_{t-1} - s_{t-1}) \quad ;t > 1$$

  - Example – accelerometer samples while walking

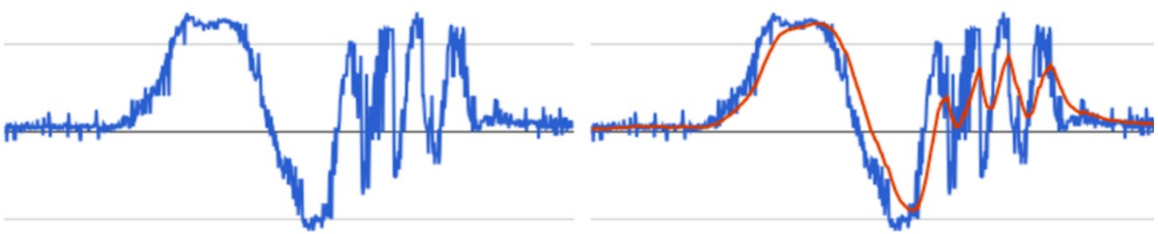



(left) accelerometer signal during walking without smoothing
(right) after exponentially weighted smoothing with smoothing = 6 (i.e. α = ⅙)

# Smoothing and Filtering (Time)

- Median filtering (if noise is sudden spikes)
  - Averaging will remove noise, but may remove data peaks and cause time lag in the data
  - Example – accelerometer samples



(a) Raw accelerometer readings (b) Exponential smoothing

$$s_1 = median(x_1, x_2, x_3)$$

$$s_2 = median(x_2, x_3, x_4)$$

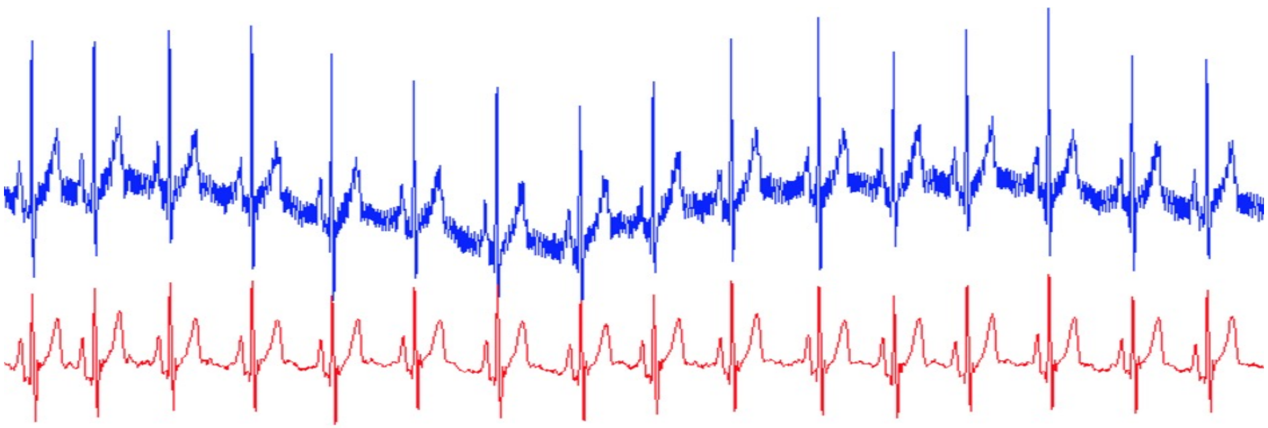
...

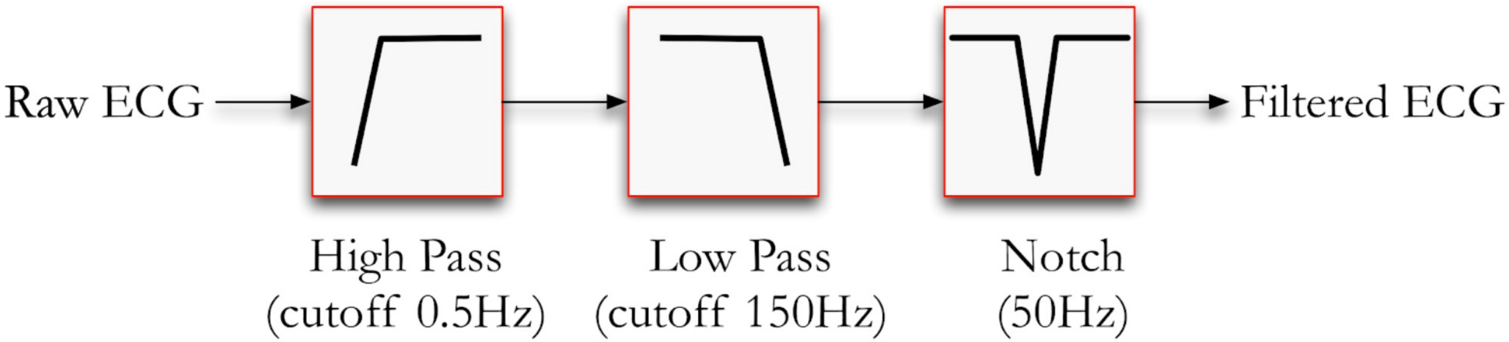
# Smoothing and Filtering (Frequency)

- Fourier transform
  - A waveform is replaced with a sum of an infinite num of sine waves, multiplied by Fourier coefficients
- Frequency domain filtering
  - Remove sins that are outside of the frequency range you are interested in (zero Fourier coefficients out)
- Filter types:
  - Low pass – block everything but low frequencies
  - High pass – block everything but high frequencies
  - Bandpass – block everything but a band of freqs.
  - Notch filter – block a very narrow band of freqs.

# Smoothing and Filtering (Frequency)

- In practice - use a chain of filters
  - Example: ECG signal filtering






University of Ljubljana
Faculty of Computer and
Information Science

# Sensing in Android

University of Ljubljana
Faculty of Computer and Information Science

# Sensors in Android

- Sensors (Android OS's def.) ≠ Sensors (our def.)
  - Sensor (android.hardware.*):
    - Motion sensors (accelerometers, gyroscopes, etc.)
    - Environmental sensors (barometers, thermometers, etc.)
    - Position sensors (orientation sensors and magnetometers)
  - Google Play Services:
    - Location – no direct access to GPS
    - Physical Activity – an already embedded classifier, no need to query acceleration, location, etc.
  - Wireless
    - Bluetooth android.bluetooth.*
    - Wi-Fi android.net.wifi.*
    - NFC andoid.nfc.*

# Sensor Framework

- SensorManager class
  - A system service for sensing management
    - e.g. infer (at runtime) which sensors are available

```
private val sensorManager =
        getSystemService(Context.SENSOR_SERVICE) as SensorManager
val deviceSensors: List<Sensor> =
        sensorManager.getSensorList(Sensor.TYPE_ALL)
```

- Sensor class
  - Sensor types as constants e.g. Sensor.TYPE_MAGNETIC_FIELD
  - Data reporting: 1) streaming or 2) on change
  - `getResolution(), getMaximumRange(), getPower()`

# Publish-Subscribe Sensing

- SensorEvent class
  - Events containing new sensed values, accuracy timestamp, and sensor type information
- SensorEventListener interface
  - Implement and override:
    - `onSensorChanged(event: SensorEvent)`
    - `onAccuracyChanged(sensor: Sensor, accuracy: Int)`
  - Register the listener

```
sensorManager.registerListener(this,
mSensor, SensorManager.SENSOR_DELAY_NORMAL)
```

  - Unregister when done, leaving the Activity/Service

```
sensorManager.unregisterListener(this)
```

# Sensor (android.hardware.*) Summary

- This is how you sense accelerometer, magnetometer, barometer, temperature sensor, light, proximity, etc.
- Before using it, verify that a sensor is present
- Unregister the listener when leaving
- Don't do heavy work in `onSensorChanged`
- Choose sensor reporting delay wisely
  - Use the highest delay that still works for your app's purpose

# Accelerometer sensing example

# Google Play Services

- A background service providing access to a range of Google's services:
  - Maps
  - Google sign in
  - Google drive
  - Location
  - Activity recognition
- Centralised handling of sensing requests reduces energy usage
  - One GPS result may be served to a bunch of apps that request location in a short time period

# Google Play Services – Location

- FusedLocationProviderClient class
  - Access to location determined via different means (GPS, network signal triangulation, etc.)
  - `getLastLocation()`
  - `requestLocationUpdates()`
    - Define request priority: from "high accuracy" to "no power", and request interval
    - Get callback when a new location info is ready
    - Don't forget to unregister the request!
- GeofencingClient class
  - Define geofence region and transition types
  - Supply Intent to be fired when the conditions are met

# Google Play Services – Location

- Permissions (request at runtime):
  - ACCESS_COARSE_LOCATION
  - ACCESS_FINE_LOCATION
  - ACCESS_BACKGROUND_LOCATION (new in API 29)
- Caution: location fetching APIs are changing all the time – check the official documentation rather than (potentially obsolete) tutorials

# Google Play Services – Activity

- A built-in classifier of physical activity (walking, cycling, still, in vehicle, running)
- ActivityRecognitionClient class
  - Subscribe to activity recognition transitions or updates
  - Supply Intent to be called when the results are ready
  - Don't forget to unregister the request!
- Permissions:
  - ACTIVITY_RECOGNITION (runtime in API 29)

# TODO

- Android Location sensing example is on Ucilnica
- Project proposal
  - Version 1 by tonight 23:59
  - Version 2 by Monday Feb 28th
- Lab
  - Attend tomorrow or complete the lab and push your code to BitBucket
    - Make sure your repository is named FRIMS2021-LAB-1
    - Make sure the TA is given access