# CS390MB:
# Mobile Health Sensing & Monitoring

Your Best Chingu

December 19, 2016

# 1 Sensor Data Smoothing & Filtering

- Sensor data is affected by **noise** - unexplained variations in data that is uninterpretable.
- Goal: Remove noise while retaining important characteristics of signal

Noise Removal Techniques can be divided into two classes:

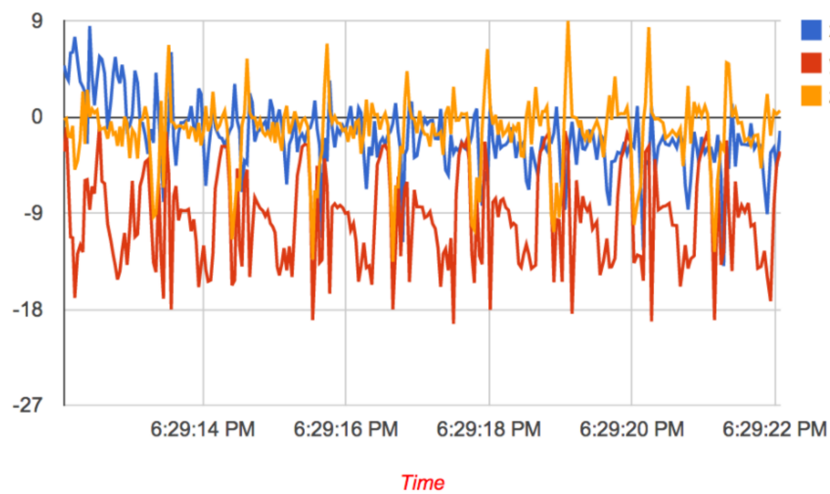
- Time-domain: intuitive way of approaching problem
- Frequency-domain: removes noise that is periodic in domain

## 1.1 Information in Signals

Two common ways for info to be represented in naturally occurring signals:

- **Time domain**: describes when something occurs and what the amplitude
- **Frequency**: domain: indirect, measures frequency, phase, and amplitude of periodic motion

## 1.2 Noisy Sensor Signals




Typical Pattern of x,y, and z accelerations while walking with smartphone

**Noise in Accelerometer Data**: categorized into two types:

**Intrinsic Sensor Noise**

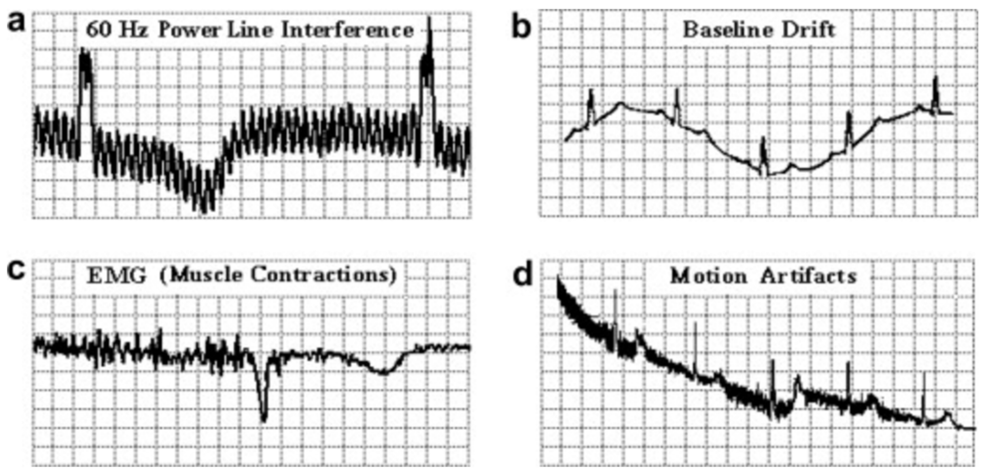
- Electronic noise from circuitry that is converting motion
- Mechanical noise of sensor

**External Vibration Noise**

- Continuous external vibrations induced by earth's movement, nearby vehicles, etc
- Tiny movements manifest as small changes in accelerometer reading
- Example: Trying to detect orientation of phone

    Vibrants cause accel. outputs to appear jittery

    Jitters need to be smoothed before applying algorithm to determine screen orientation



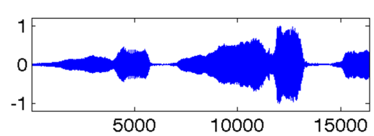

Typical ECG signal with different interference sources

### Noise in Electrocardiogram (ECG) Data

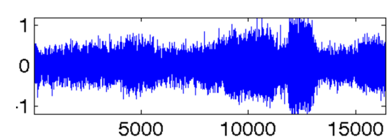
- Problem: power line interference: 50Hz signal causes electromagnetic interference
- Issue is problematic for low frequency signals like ECG
- Other noise: breathing, muscle contractions, body movements, etc

### Noise in Image Data

- Noise is often caused by camera
- Poor illumination conditions, high temperature, electronic noise in circuit



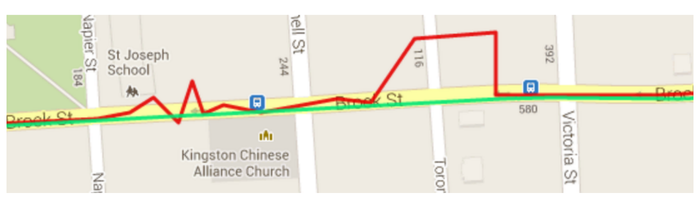

(a) Original Audio Signal




(b) Noisy Audio Signal

### Noise in Audio Signals

- Noise could be due to ambient sound or loud noise nearby such as a construction site
- Hardware and circuit could add to noise




Noisy GPS Readings while driving (red), Actual trajectory (green)

### Noise in GPS Data

- Noise due to clock error, multipath effects due to buildings, weather conditions
- Raw data coming from GPS receiver has noise that is being smoothed before display

## 1.3 Time-series Smoothing and Filtering

### Over-sampling and Averaging

- Many sources of noise are random: has roughly equal amounts of positive and negative changes
- Noise is uncorrelated in time, has zero mean, and finite variance
- Noise can be reduced by **oversampling** the sensor and **averaging** the values
- Example:
  Can use a sampling rate of 100 Hz
  Average every 10 samples readings
  Report average value at 10Hz frequency
- If we have $n$ samples of a random noise signal and average them, we reduce noise by a factor of $1/\sqrt{n}$

### Moving Average Smoothing

- Replace each sample by average of current sample, sample before, and sample after
- Let us represent input accelerometer signal as follows: $x = x_1, x_2, ..., x_n$ where index is sample number
- Output of moving average filter:

$$
\begin{aligned}
s_1 &= (x_1 + x_2 + x_3)/3 \\
s_2 &= (x_2 + x_3 + x_4)/3 \\
s_3 &= (x_3 + x_4 + x_5)/3 \\
s_{n-2} &= (x_{n-2} + x_{n-1} + x_n)/3
\end{aligned}
$$

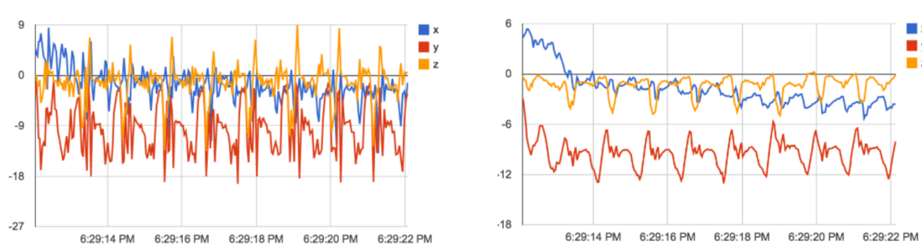
- Increasing smoothing window will make signal look cleaner and visually pleasing
- However, too large a window can smooth out important characteristics of signal
- Moving Average assumes random noise

### Exponential Smoothing

- Effective when noise is time varying
- Similar to moving average, but assigns exponentially decreasing weights as observations get older
- Recent observations are given relatively more weight than older observations
- smoothing factor $\alpha$: $0 < \alpha < 1$,
- Smoothed output $s_t$: weighted average of current observation $x_t$ and previous smoothed output $s_{t-1}$

$$
\begin{aligned}
s_1 &= x_0 \\
s_t &= \alpha x_{t-1} + (1-\alpha)s_{t-1} \\
&= \alpha x_{t-1} + \alpha(1-\alpha)x_{t-2} + (1-\alpha)^2 s_{t-2} \\
&= \alpha[x_{t-1} + (1-\alpha)x_{t-2} + (1-\alpha)^2 x_{t-3} + (1-\alpha)^3 x_{t-4} + ...] + (1-\alpha)^{t-1} x_0
\end{aligned}
$$

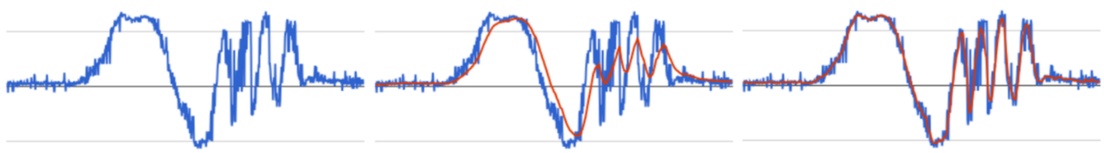
- Larger values of $\alpha$ reduce the level of smoothing
- $\alpha = 1$ is same as original series (with lag of one time unit)




(a) Signal during walking without smoothing (b) After exponentially weighted smoothing

### Median Filtering

- When noise appears like sudden spikes, moving average and exponential smoothing are not appropriate
- Exponential smoothing will average some peaks in data and don't have same amplitude and has lag



(a) Raw accelerometer readings (b) Exponential smoothing (c) Median filtering

Median filtering is better for removing salt-and-pepper noise than exponential smoothing

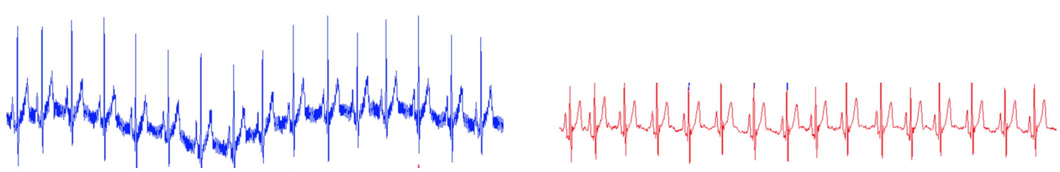
- Operates over sliding windows like moving average and exponential smoothing
- Computes median over each window rather than average
- If input accelerometer signal is $x = x_1, x_2, ..., x_n$, output of median filter is:

$$
\begin{aligned}
s_1 &= \text{median}(x_1, x_2, x_3) \\
s_2 &= \text{median}(x_2, x_3, x_4) \\
&... \\
s_{n-2} &= \text{median}(x_{n-2}, x_{n-1}, x_n)
\end{aligned}
$$

## 1.4 Frequency-domain Filtering

- Low-pass filter lets low-freq components below threshold through while removing high freq components
- High-pass filter does reverse and lets high freq components through while removing low freq components
- Notch filter removes a specific frequency from signal

### ECG Noise Removal



(a) ECG signal with baseline wander, powerline interference, and high-freq noise

(b) Filtered ECG Signal

**Baseline Wander**: low-frequency component present in ECG system which causes signal to "wander" off from actual ECG waveform

- Due to offset voltages in electrodes, periodic breathing, body movement
- In figure, baseline wander is slowly oscillating waveform with much lower frequency than ECG signal
- Can be removed by using high-pass filter with cutoff to remove baseline wander

**Powerline Noise**: frequency of alternating current in electrical mains is around 50-60Hz.

- Can be removed from ECG signal with notch filter at 50/60Hz

**High Frequency Noise**: Pacemakers, phones, other electronics are sources of high frequency noise

- Removed with low-pass filter

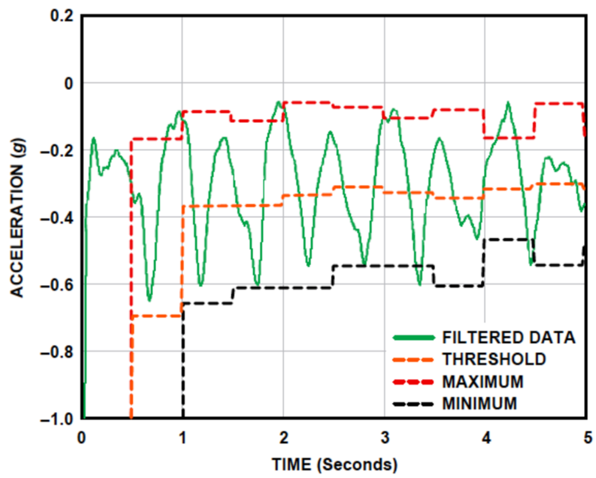
# 2 Designing a Pedometer

Acceleration changes as a result of step and can result in changes along all x, y, and z-axes.
Goal: design an *orientation-independent* algorithm

## 2.1 Step Detection Algorithm

**Smoothing**

- Average nearby values to remove noise
- Increasing smoothing windows creates cleaner signal, but may smooth out steps




Filtered data on most active axis

**Dynamic Detection Threshold**

- No fixed threshold that we can use since threshold depends on orientation of accelerometer
- Need to use dynamic thresholding scheme to detect a step

1. Keep track of axis along which maximum acceleration occurs
2. Keep track of min and max acceleration levels over a window of samples
3. Average value: (Max + Min)/2 is **dynamic threshold level** and detects steps for next window sample

**Step Detection Algorithm**

- **Step** is detected if there is a **negative slope** of acceleration plot when the acceleration curve crosses below the dynamic threshold

**Periodicity**

- When pedometer vibrates very rapidly or slowly, the step counter will take it as a step
- Invalid vibrations must be discarded

First Approach: Look at time period between any two steps

- Assume people run as fast as five steps per second and walk as slow as one step every two seconds
- Interval between two valid steps is in range (0.2 to 2.0s)
- Steps outside time window is discarded

Second Approach: Look for periodic walking pattern

- Look at time between steps and see if duration repeats (with small variations)
- Repeated pattern suggests person is walking

# 3 Activity Recognition using Inertial Sensors

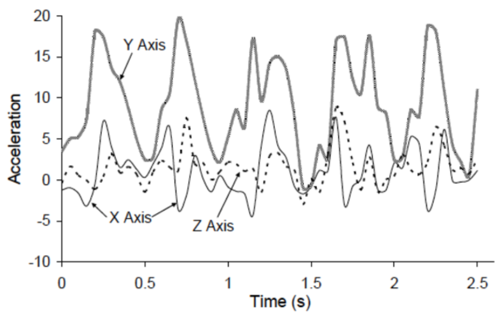
## 3.1 Detection vs. Classification

- With activity recognition, we assume we don't know distinguishing characteristics of each activity
- Provide training data: sample datasets of each class
- Provide features: large set of possible characteristics of data that may be important
- Let automated algorithm identify what features are most useful to distinguish between classes

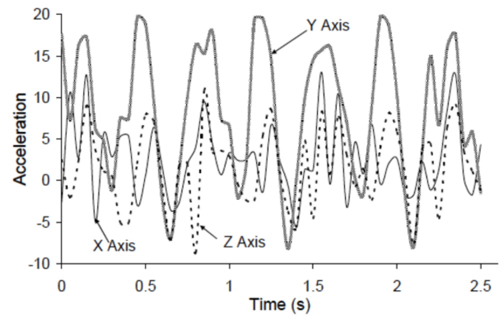
## 3.2 Labeled Data Collection

- Labels: ground truth corresponding to raw data must be available
- Training data is used to develop classification algorithm
- Carrying phone in different orientations will help algorithm to be less sensitive to orientation variations

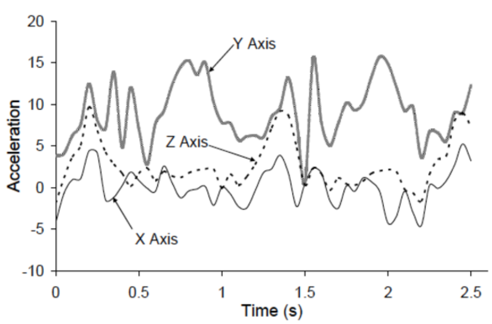
## 3.3 Visualizing Common Activities



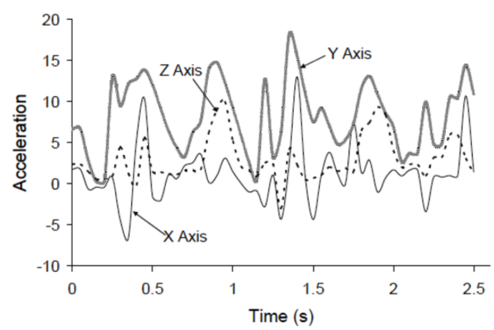

**(a) Walking**



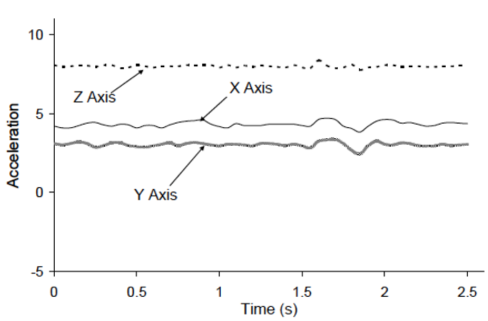

**(b) Jogging**



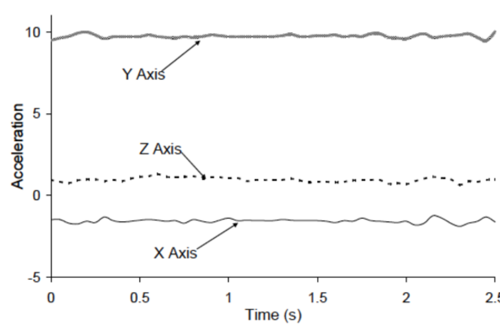

**(c) Ascending Stairs**




**(d) Descending Stairs**




**(e) Sitting**



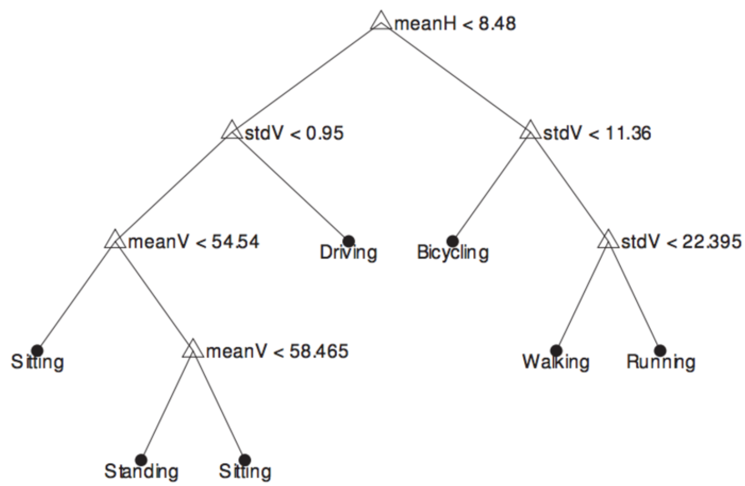

**(f) Standing**

### 3.4 Feature Generation & Data Transformation

- Distinguishing features: frequency, frequency changes
- Useful to divide features into two classes: a) time domain features b) frequency domain features

| Time Domain Features | Frequency Domain Features |
|---|---|
| Mean, Median, Variance, Standard Deviation | Dominant frequency, Signal Energy |
| Min, Max, Range, Zero-crossings, Angle, Angular velocity | |

### 3.5 Decision Tree Classifier




**Building Decision Tree**

- At each node of tree, choose attribute of data that most effectively splits set of samples into subsets enriched in one class or other
- Splitting criterion: **information gain** - metric for describing how much separation is achieved after split compared to before

**Entropy**

- A way to measure impurity (0 = minimum impurity, 1 = maximum impurity)
- Let $p_i$ be probability of class $i$ - compute as proportion of class $i$ in set

$$\text{Entropy} = \sum_i -p_i \log_2 p_i$$

**Information Gain**

- Tells us how important a given attribute of the feature vectors is

$$\text{Information Gain} = \text{entropy(parent)} - [\underline{\text{weighted}} \text{ average entropy (children)}]$$

**Pseudocode**

1. For each attribute $a$, find normalized information gain from splitting on $a$
2. Let *a_best* be attribute with highest normalized information gain
3. Create decision node that splits on *a_best*
4. Recurse on sublists obtained by splitting on *a_best* and add nodes as children of node

# 4 Evaluating Classifier Performance

## 4.1 Cross Validation

### Holdout Method

- Data set is separated into two sets: a) training set b) testing set
  common rule: 70% of dataset for training and 30% for testing
- Dividing data into subsets is done randomly to guarantee no systematic error
- Classifier is learnt using training set only
  predicts output values for data in testing set
- Errors are accumulated before to give mean absolute test set error

### N-fold Cross Validation

- Data set is divided into $n$ subsets and holdout method is repeated $n$ times
- Each time, one of $n$ subsets is used as test and other $n-1$ subsets form training set
- Average error across all $n$ trials is computed
- Advantage: matters less how data gets divided, more robust
- Disadvantage: training algorithm is rerun $n$ times

**Overfitting**: phenomenon of relying on patterns that are strong only in training data

## 4.2 Confusion Matrix: Performance Measures

- Rows correspond to known class of data
- Columns correspond to predictions made by model
- Diagonal elements show number of correct classifications made for each class
- Off-diagonal elements show errors made

$TP$ and $FP$ are numbers of true positive and false positive predictions for considered class.

**Accuracy**: overall correctness of model

- Sum of correct classifications divided by total number of classifications

**Precision**: measure of accuracy provided that a specific class has been predicted (cell / sum of column)

$$\text{Precision} = \frac{TP}{TP + FP}$$

**Recall**: measure of ability of model to select instances of a certain class from a data set (cell / sum of row)

$$\text{Recall} = \frac{TP}{TP + FN}$$

**F-measure**: weighted average of precision and recall (best at 1, worst at 0)

$$\text{F} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

# 5 Quantified Self & Personal Data Analytics

## 5.1 Quantified Self Movement

**Quantified self** is a grassroots movement where people are measuring, logging, and sharing metrics related to their physical and mental health.

## 5.2 Obtaining Data About Yourself

- Logging data from digital traces (Fitbit, Airline sites, Outlook, etc)
- Process is not completely monitored (yet)

## 5.3 Analyzing Lifelog Data for Useful Insights

**Visualizing Trends in Data**

- Charting and plotting trends in each variable

**Mining Patterns in Data**

- Association rule mining: useful when you have sequences of labeled data
  Ex: data about food locations visited, attentiveness in class, productivity level, etc
- Use association rule mining to discover association between food habits and other indicators

**Identifying Predictors**

- Want to identify predictors for a behavior or variable
- Example: record weight gain data with checkins for #visits to coffee shop, #visits to Chipotle, etc
- Linear regression: assumes form for relationship between predictors $X_i$ and response variable $y$
  $\beta$ represents weight associated with specific predictor

$$y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \alpha$$

- After computing coefficients, next Q: how well have we captured trends present in weight gain?
  There may be some other variable affecting weight gain
- Look at residual error after fitting and see if there is any pattern in residual error
  If there is a pattern, there is a trend we did not capture using set of predictor variables
  Or relationship between weight and predictive variables is not linear
- Next step: figuring out what predictors matter
- Need a hypothesis test for statistical significance
  Measure: $p$-value
    Low $p$-value $< 0.05$ means variable is highly predictive
    High $p$-value means not useful

**Eliciting Change in Behavior**

- Goal of self-tracking: fixing a problem and discovering patterns and roots of problem

Experimental Design: A = baseline, B = treatment

- **AB** : simplest and weakest of all in capturing causality
- **ABA**: captures changes in $Y$ before and after treatment
  Helps conclude if treatment works and how long effects last
- **ABAB**: useful to capture intensity of treatment is associated with intensity of outcome
  Can compare changes in both treatment phases
  In either treatment phases, can replace one with placebo

# 6 Voice-based Health Analytics

## 6.1 Voice Analysis Library: Feature Set

### Mel-frequency Spectral Coefficients (MFCC)

- Extracts features closest to human perception of voice
- Relates perceived frequency of a pure tone to its actual measured frequency
- Log of Mel filterbank: human hearing - we don't hear loudness on a linear scale
  To double perceived volume of sound, we need to put 8 times as much energy into it

### Other Audio Features

**Prosody**: sound characteristics such as syllable length, loudness, and pitch

- **Pitch**: describes how listener perceives a sound
  Sudden increase in pitch $\rightarrow$ high activation, anger
  Low variance of pitch $\rightarrow$ low energy, sadness
- **Intensity**: reflects effort to produce speech
  Rapid rise of energy $\rightarrow$ angry utterance
  Low intensity $\rightarrow$ sad speech
- **Temporal Aspects**: describes speech rate and voice activity (pauses)
  Can reflect emotions
- **Voice Quality**: emotions influence voice quality of utternaces
  Sharp/jagged vs soft
  Glottal waveforms are useful
  Sudden change in air flow produces high frequency
- **Spectogram**: describes energy distribution across frequency bands
  Certain frequency may be speaker dependent
  Used to reflect emotions
- **Other Statistical Measures**: can help represent all possible dynamics affected by emotions

## 6.2 Voice Analysis Library: Classification

### Speech Processing

- Human speech can be broken into phenomes
- Challenge in speech recognition: recognizing sequence of phenomes as particular word
- Speech recognizers use a HIdden Markov Model (HMM)

### Diagnosis of Mental Illness

- Aspects of speech help describe patient's state of mind under domains of behavior, cognition, etc.
- Depressed patients express slow responses, monotonic phrases, and poor articulation
- Agitated behavior includes expansive gesturing, pacing
- Gaussian Mixture Model: clustering approach for classifying audio features

## 6.3 Monitoring Affect with a Mobile Phone

- Monitoring stress in everyday lives: phones can monitor stress and inform ways to de-stress
- Monitoring social interactions (or lack of it)

# 7 Physiological Sensing

## 7.1 Electrocardiogram (ECG)

**ECG**: recording of electrical activity of heart

- Each heartbeat: electrical signal spreads from top to bottom of heart
- Can measure heart's electrical signals by:
  Placing two electrodes at different points on chest
  Measuring electrical activity between electrodes

ECG can show:

- How fast heart is beating
- Rhythm of heartbeat (steady vs irregular)
- Strength and timing of electrical singals

**Detecting Peaks and Troughs in ECG Waveform**

- Can use peak detection for step detector
- Look for change in slope and sequence info to label appropriate peaks

**Extracting ECG Features**

- Once we get 5 or 6 peaks and troughs, timing differences between each is useful for classification
- RR interval corresponds to time between two successive heartbeats
- Computing heartrate: $HR = 60/RR$

## 7.2 Photoplethysmography (PPG)

**PPG**: non-invasive technique for measuring blood volume changes in vessels close to skin

- Place finger over camera with flash
- Camera records light absorbed by finger tissue
- Each frame is processed by splitting every pixel into RBH components
- Components are processed to extract HR and breathing rate

**Extracting Heart Rate from PPG**

- Changes in arterial blood volume correspond to heart rate
- When blood retracts, more light passes through tissue

- Green intensity average in PPG signal forms peaks corresponding to cardiac pulse
- Peak: highest average of green values in fixed window size
- After peak detection: compute time difference between consecutive peaks
- Time difference = RR interval, $HR = 60/RR$

**Extracting Breathing Rate from PPG**

- Respiratory Sinus Arrhythmia (RSA): naturally occurring variation in heart rate that occurs in breathing cycle
- Herat rate increases during inspiration and decreases during expiration
- We look for frequency of changes in heart rate
- Use FFT to convert from time to frequency domain and take dominant frequency from FFT

### 7.3 Electrodermal Activity (EDA)

**EDA**: measures electrical chnages at skin surface that arise when skin receives innervating signals from brain

- When people experience emotional arousal, increased cognitive workload or physical exertion:
  Brain sends signals to skin to increase level of sweating
- EDA results from sympathetic neuronal activity - neural response cannot be controlled consciously
- Used to examine implicit emotional responses and for lie detection

**EDA Features**: Two main components:

- Skin Conductance Level (SCL): slower acting components:
- Skin Conductance Response (SCR): faster changing components
  Example: if startled, SCR changes in seconds while SCL changes over minutes

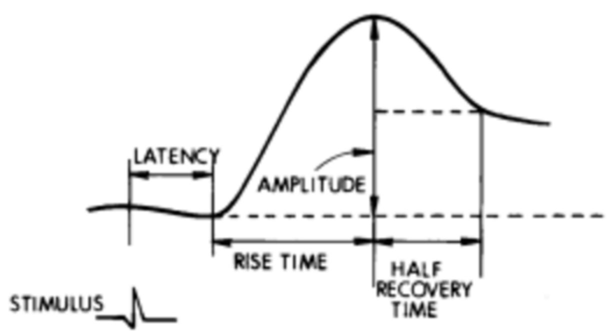



Figure 5. Graphical representation of principal EDA components.

Typical SCR

SCR can be sub-divided into features useful for classification

- **Latency**: amount of time between stimulus and rise of wave
- **Rise time**: time for skin conductance to shoot up to peak
- **Amplitude**: height of the SCR
- **Half recovery time**: amount of time it takes for wave to fall back to half its amplitude

SCL is background signal in absence of SCR

- Select small window of EDA samples with no SCR
- SCL level is average value of EDA

# 8 GPS Clustering & Analytics

## 8.1 Clustering Location Data

- **Noise**: GPS error depends on numbers factors: satellites, tall buildings, indoor/outdoor
- **Meaningful Clusters**: identifying which GPS coordinates correspond to meaningful clusters
- **Semantic Location**: converting clusters to places (home, work, coffee shop)

## 8.2 GPS Clustering

**Phase I: Pre-processing**: removing noise and outliers

1. Remove low density points
2. Remove points with movement
3. Reduce data for stationary locations

**Phase II: Clustering**

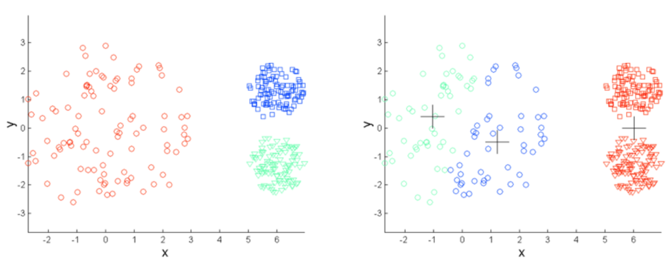
**K-means**: key parameter is $k$, number of clusters

Given k, the k-means algorithm consists of iterative algorithm with 4 steps:

- Select $k$ initial centroids at random from points
- **repeat**
    Assign each object to cluster with nearest centroid (in terms of distance)
    Re-compute each centroid as mean of objects assigned to it
- **until** centroids do not change

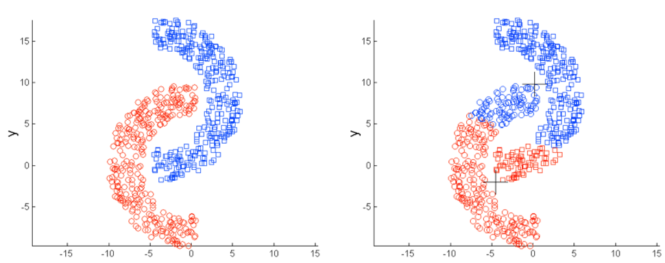
Simple and effective, but has **limitations**

- **Differing sizes**: K-means assumes that cluster are roughly similarly sized
- **Differing density**: K-means relies on centroid of points to separate into clusters



Original Points    K-means (3 Clusters)

- **Non-globular shapes**: K-means cannot deal with irregular and skewed shapes

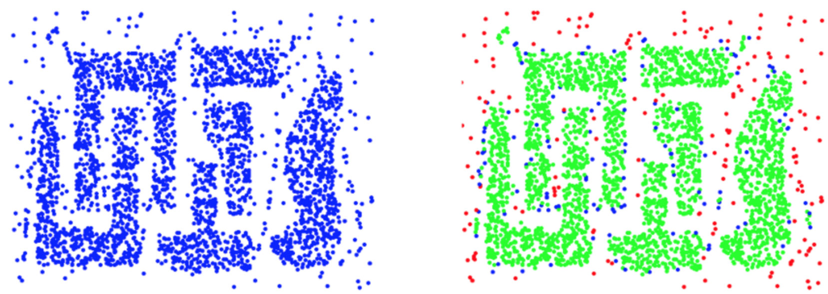


Original Points    K-means (2 Clusters)

**DBSCAN**: assumes cluster is a connected regions where points are relatively dense
Requires two parameters: a) $\epsilon$ (eps) b) minPts

Given parameters, points can be separated into three classes:

- Point is **core point** if it has more than minPts within $\epsilon$
- **Border point** has fewer than minPts within $\epsilon$ but is in neighborhood of core point
- **Noise point** any point that is not a core point or border point



**Original Points** **Point types: core, border and noise**

**Eps = 10, MinPts = 4**

Once points are divided, DBScan algorithm:

- Removes all noise points
- Performs clustering on remaining points in iterative manner

**Pseudocode**

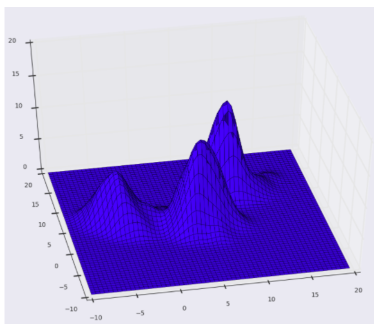
```
DBSCAN(D, eps, MinPts)
    C = 0
    for each point P in dataset D
        if P is visited
            continue next point
        mark P as visited
        NeighborPts = regionQuery(P, eps)
        if sizeof(NeighborPts) < minPts
            mark P as NOISE
        else
            C = next cluster
            expandCluster(P, NeighborPts, C, eps, MinPts)

expandCluster(P, NeighborPts, C, eps, MinPts)
    add P to cluster C
    for each point P' in NeighborPts
        if P' is not visited
            mark P' as visited
            NeighborPts' = regionQuery(P', eps)
            if sizeof(NeighborPts') ≥ MinPts
                NeighborPts = NeighborPts ∪ NeighborPts'
        if P' is not yet a member of any cluster
            add P' to cluster C

regionQuery(P, eps)
    return all points within P's eps-neighborhood (including P)
```

**Mean Shift**: considers feature space as probability density function

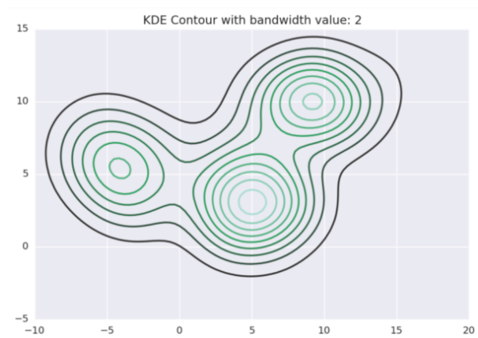
- Does not require prior knowledge of number of clusters and does not constrain cluster shape
  Ideal for handling clusters of arbitrary shape and number
- Places a kernel on each point in data set
  kernel: a weighting function
- Adding all of individual kernels up generates a probability surface
- Primarily a mode finding algorithm - number of clusters is obtained by number of modes

**High Level Intuition**:

1. Fix window around each data point
2. Compute mean of data within window
3. Shift window to mean and repeat till convergence



(a) KDE surface plot




(b) KDE contour plot

**Phase III: Clusters to Semantic Locations**

- Can use reverse geo-coding on cluster centroids
- Going from GPS coordinates of centroid to name of place