STATISTICAL SIGNAL ANALYSIS

Dr TIAN Jing

tianjing@nus.edu.sg

Module objective

Module: Statistical signal analysis

Knowledge and understanding

- Understand the fundamentals of statistical signal analysis, learning distribution from signal, and statistical classification

Key skills

- Design, build, implement a statistical classification approach for signal features

Statistical signal analysis

- Learning: For a given set of observations $\boldsymbol{X} = [\boldsymbol{X}_1, \boldsymbol{X}_{2,}, \cdots, \boldsymbol{X}_N]$ from some random variables, learn a model $f(\boldsymbol{X}|\boldsymbol{\theta})$ (defined by the parameter $\boldsymbol{\theta}$) to best describe the data.
- Prediction: Assume that we have two classes of objects as ω_1 and ω_2, and we already have learned models $f(\boldsymbol{X}|\boldsymbol{\theta_1})$ and $f(\boldsymbol{X}|\boldsymbol{\theta_2})$ for ω_1 and ω_2 , respectively. For a new observation $\boldsymbol{X}_n$, predict which class it belongs to, that is $\boldsymbol{f}(\omega_1, \omega_2|\boldsymbol{X}_n, \boldsymbol{\theta_1}, \boldsymbol{\theta_2})$.

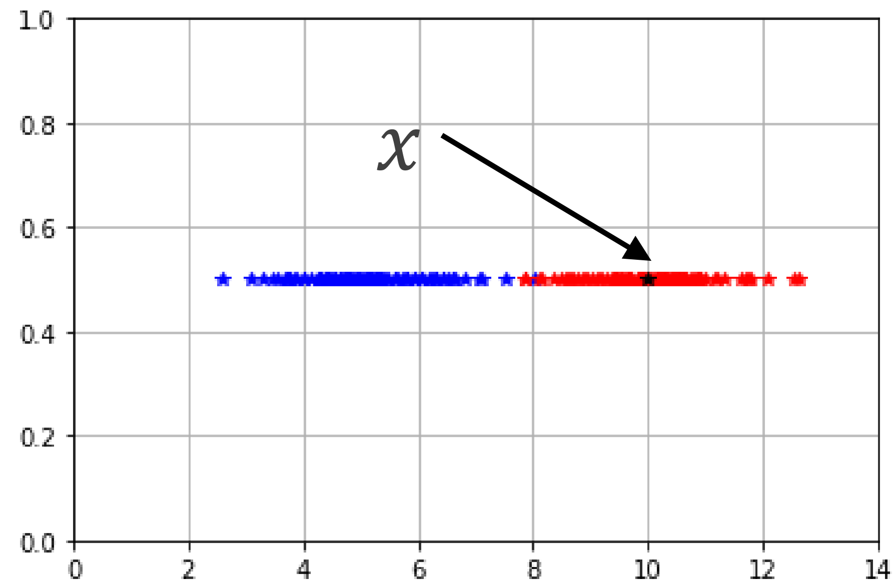
Signal classification

- Given a signal x, two categories of data, say ω_1, ω_2, does the signal x belong to ω_1 or ω_2?
- That means, we need to evaluate $P(\omega_1|x)$ and $P(\omega_2|x)$ and make decision (next slide).
- The class posterior probability is

Likelihood Prior

$$P(\omega_i|x) = \frac{P(x|\omega_i)P(\omega_i)}{P(x)}$$

Evidence



Signal classification

- Class priors $P(\omega_i)$
  - How much of each class? $P(\omega_i) \approx N_i/N$
- Class likelihood $P(x|\omega_i)$
  - Requires that we have a model for each ω_i, for example, ω_i can be modelled as Gaussian distribution
- Evidence
  - $P(x) = P(x|\omega_1)P(\omega_1) + P(x|\omega_2)P(\omega_2)$
- Classification rule
  - If $P(\omega_1|x) \geq P(\omega_2|x)$, then x is classified as ω_1
  - If $P(\omega_1|x) < P(\omega_2|x)$, then x is classified as ω_2

Example: Naive Bayes classifier

Given: $\boldsymbol{x}$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
Predict: PlayTennis Yes or No?

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example

Given: $\boldsymbol{x}$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
Predict: PlayTennis Yes or No?

Bayesian Rule	
P(*Yes*|$\boldsymbol{x}$)	0.0053
[P(*Sunny*|*Yes*)P(*Cool*|*Yes*)P(*High*|*Yes*)P(*Strong*|*Yes*)]P(Play=*Yes*)	
P(*No*|$\boldsymbol{x}$)	0.0206
[P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*)]P(Play=*No*)	

Decision: Given the fact $P(Yes|\boldsymbol{x}) < P(No|\boldsymbol{x})$, we decide $\boldsymbol{x}$ to be *No*.

P(Outlook=*Sunny*|Play=*Yes*) = 2/9
P(Temperature=*Cool*|Play=*Yes*) = 3/9
P(Huminity=*High*|Play=*Yes*) = 3/9
P(Wind=*Strong*|Play=*Yes*) = 3/9
P(Play=*Yes*) = 9/14

P(Outlook=S*unny*|Play=*No*) = 3/5
P(Temperature=*Cool*|Play==*No*) = 1/5
P(Huminity=*High*|Play=*No*) = 4/5
P(Wind=*Strong*|Play=*No*) = 3/5
P(Play=*No*) = 5/14

Details, refer to next slide

Example

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

Total: $P(Play = Yes) \ = \ 9/14, P(Play = No) \ = \ 5/14$

How about continuous data?

For the training data

- Calculate the prior probability by counting the number of data in each class. $P(Window\ Glass)$ and $P(Not\ Window\ Glass)$
- Build the likelihood probability distribution, for each class, assume each attribute (column) follows Gaussian distribution, calculate its mean and variance.

For the test data

- Apply the same formula. Calculate the likelihood probability using the built Gaussian functions, $P(Refractive\ index = 1.71720|Window\ Glass)$ and $P(Refractive\ index = 1.71720|Not\ Window\ Glass)$

	No.	Refractive index	Sodium	Magnesium	Aluminum	Silicon	Potassium	Calcium	Barium	Iron	Class
Training data	1	1.51824	12.87	3.48	1.29	72.95	0.60	8.43	0.00	0.00	Window Glass
	2	1.51832	13.33	3.34	1.54	72.14	0.56	8.99	0.00	0.00	Window Glass
	3	1.51747	12.84	3.50	1.14	73.27	0.56	8.55	0.00	0.00	Window Glass
	4	1.51775	12.85	3.48	1.23	72.97	0.61	8.56	0.09	0.22	Window Glass
	5	1.51768	12.65	3.56	1.30	73.08	0.61	8.69	0.00	0.14	Window Glass
	6	1.51769	12.45	2.71	1.29	73.70	0.56	9.06	0.00	0.24	Window Glass
	7	1.51892	13.46	3.83	1.26	72.55	0.57	8.21	0.00	0.14	Window Glass
	8	1.51727	14.70	0.00	2.34	73.28	0.00	8.95	0.66	0.00	Not Window Glass
	9	1.51545	14.14	0.00	2.68	73.39	0.08	9.07	0.61	0.05	Not Window Glass
	10	1.51994	13.27	0.00	1.76	73.03	0.47	11.32	0.00	0.00	Not Window Glass
Test data	100	1.51720	13.38	3.50	1.15	72.85	0.50	8.43	0.00	0.00	

Learning distributions for easy data

- Problem: Given a collection of examples, estimate its single distribution model
- Solution:
  - Assign a model to the distribution
  - Learn parameters of model from data

Bernoulli distribution

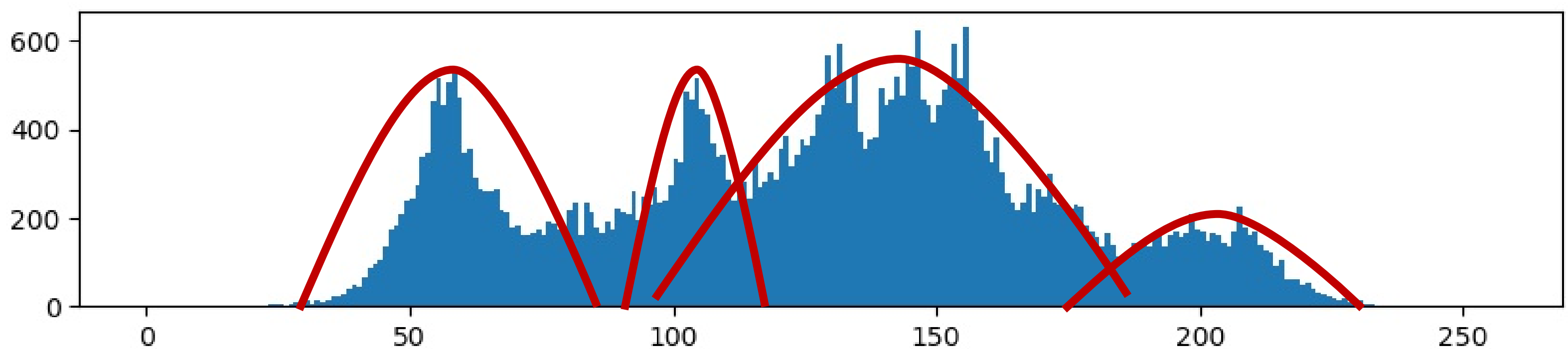
$$p(\mathbf{x}|\boldsymbol{\mu}) = \prod_i \mu_i^{x_i}(1-\mu_i)^{x_i}$$

Gaussian distribution

$$p(\mathbf{x}|\boldsymbol{\mu},\Sigma) = \frac{1}{\sqrt{(2\pi)^N|\Sigma|}} exp\left\{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})\Sigma^{-1}(\mathbf{x}-\boldsymbol{\mu})\right\}$$

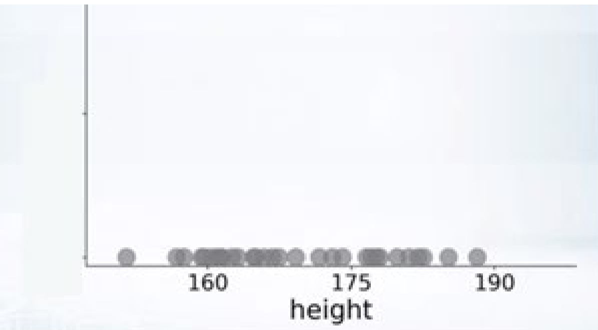
How about complicated data

- Example: The dataset (illustrated in figure below) is complicated so that we can not use a single function (e.g., single Gaussian function) to fit data.
- We can fit the following dataset using several (say, 4) Gaussian functions. How do we know which data is used to fit which Gaussian function?
- We need to perform data clustering (data labelling) and distribution estimation together.

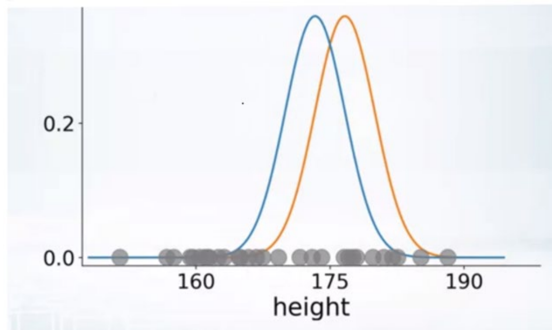


Example: Clustering

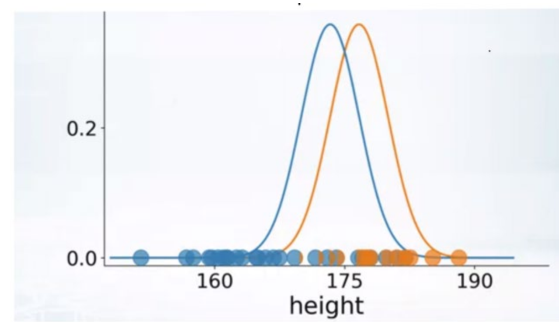
Dataset: Students' height



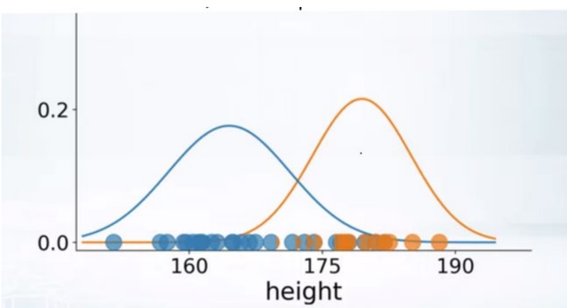
Step1: (randomly) Initialize distribution



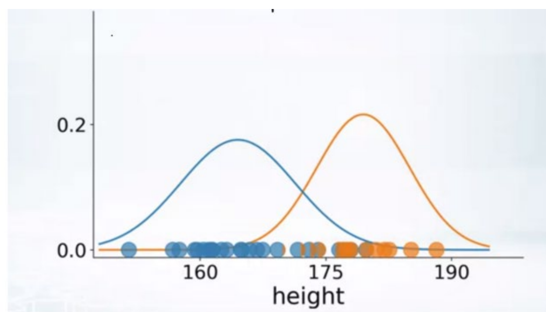
Step2: Assign data label



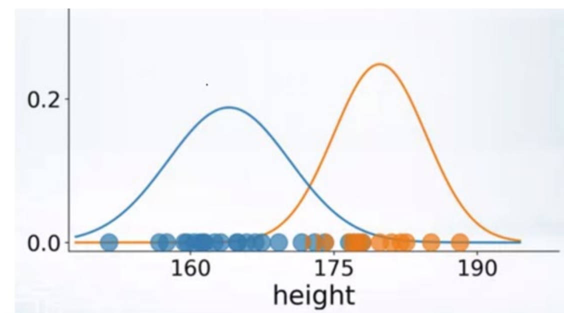
Step3: Update distribution



Step4: Update data label



Step5: Update distribution



Photos: https://towardsdatascience.com/gaussian-mixture-models-d13a5e915c8e

Learn model parameters: A toy task

Example: 6 3 1 5 4 1 2 4 …

- A person shoots a loaded dice repeatedly
- You observe the series of outcomes
- You can form a good idea of how the dice is loaded, by finding what the probabilities of the various numbers are for dice
- $\text{Probability}(number) = \dfrac{\text{Count}(number)}{\text{Sum}(rolls)}$

Learn model parameters: A complicated task with additional rules

- Additional rules
  - Two dice (say, **yellow dice** and **blue dice**) are available.
  - The dice are differently loaded for the two of them.
  - We observe the series of outcome.
  - There is a "caller" who randomly calls out the outcomes.
  - At any time, you do not know which of the two dice you are calling out,
- <u>How do you determine the probability distributions for the two dice?</u> If you do not even know what fraction of time the blue numbers are called, and what fraction are yellow.

Use case: You are given a set of GPA scores for students in TWO classes (AI and DS). You are asked to build the distribution of GPA scores of each class. But you don't know the label (the student belongs to AI or DS class) of individual GPA score.

Key idea: Introduce a hidden label

Objective: We need to formulate following distributions by filling these two tables.

X	1	2	3	4	5	6
$P(X\|\text{Blue})$						
$P(X\|\text{Yellow})$						

$P(Z=\text{Blue})$	
$P(Z=\text{Yellow})$	

- The caller will call out a number X IF
  - He selects "Yellow", and the Yellow dice rolls the number X

  OR
  - He selects "Blue" and the Blue dice rolls the number X
- $P(X) = P(\text{Yellow})P(X|\text{Yellow}) + P(\text{Blue})P(X|\text{Blue})$
  - E.g. $P(6) = P(\text{Yellow})P(6|\text{Yellow}) + P(\text{Blue})P(6|\text{Blue})$

He selects Yellow Dice rolls number 6 He selects Blue Dice rolls number 6

Solution: Expectation maximization

- Iterative solution

- Get some initial estimates for all parameters
  - Dice shooter example: This includes probability distributions for dice AND the probability with which the caller selects the dice

- Two steps that are iterated:
  - Expectation Step: Estimate the values of unseen variables
  - Maximization Step: Using the estimated values of the unseen variables as truth, estimates of the model parameters

Expectation maximization

Step 1: Initialization

- We (guess) obtain an initial estimate for the probability distribution of the two sets of dice:

X	1	2	3	4	5	6
$P(X\|\text{Blue})$	0.3	0.3	0.1	0.1	0.1	0.1
$P(X\|\text{Yellow})$	0.4	0.05	0.05	0.05	0.05	0.4

- We (guess) obtain an initial estimate for the probability with which the caller calls out the two shooters

$P(Z = \text{Blue})$	0.5
$P(Z = \text{Yellow})$	0.5

Expectation maximization

Step 2: Estimate hidden labels

- Every observed roll of the dice contributes to both "Yellow" and "Blue"

6 4 5 1 2 3 4 5 2 2 1 4 3 4 6 2 1 6

Probability to be Yellow	Probability to be Blue
6 (0.8), 4 (0.33), 5 (0.33), 1 (0.57), 2 (0.14), 3 (0.33), 4 (0.33), 5 (0.33), 2 (0.14), 2 (0.14), 1 (0.57), 4 (0.33), 3 (0.33), 4 (0.33), 6 (0.8), 2 (0.14), 1 (0.57), 6 (0.8)	6 (0.2), 4 (0.67), 5 (0.67), 1 (0.43), 2 (0.86), 3 (0.67), 4 (0.67), 5 (0.67), 2 (0.86), 2 (0.86), 1 (0.43), 4 (0.67), 3 (0.67), 4 (0.67), 6 (0.2), 2 (0.86), 1 (0.43), 6 (0.2)

Recall that we randomly guess in Step 1

$$P(Z = \text{Blue}) = P(Z = \text{Yellow}) = 0.5$$
$$P(6|\text{Blue}) = 0.1, P(6|\text{Yellow}) = 0.4$$

For roll number 6, we estimate its label

$$P(\text{Yellow}|X = 6) = P(X = 6|Z = \text{Yellow})P(Z = \text{Yellow}) = 0.4 \times 0.5 = 0.2$$
$$P(\text{Blue}|X = 6) = P(X = 6|Z = \text{Blue})P(Z = \text{Blue}) = 0.1 \times 0.5 = 0.05$$

After normalization, we have $P(\text{Yellow}|X = 6) = 0.8, P(\text{Blue}|X = 6) = 0.2$

Expectation maximization

Step 2: Estimate hidden labels

- Every observed roll of the dice contributes to both "Yellow" and "Blue"
- Total count for "Yellow" is the sum of all the posterior probabilities in the Yellow column: 7.31
- Total count for "Blue" is the sum of all the posterior probabilities in the Blue column: 10.69

Called	$P(\text{Yellow}\|X)$	$P(\text{Blue}\|X)$
6	.8	.2
4	.33	.67
5	.33	.67
1	.57	.43
2	.14	.86
3	.33	.67
4	.33	.67
5	.33	.67
2	.14	.86
2	.14	.86
1	.57	.43
4	.33	.67
3	.33	.67
4	.33	.67
6	.8	.2
2	.14	.86
1	.57	.43
6	.8	.2
	SUM=7.31	SUM=10.69

Expectation maximization

Step 3: Update $P(X|\text{Yellow})$

- Total count for "Yellow": 7.31
  - Total count for 1: 1.71
  - Total count for 2: 0.56
  - Total count for 3: 0.66
  - Total count for 4: 1.32
  - Total count for 5: 0.66
  - Total count for 6: 2.4
- Updated probability of Yellow dice:
  - $P(1 \mid \text{Yellow}) = 1.71/7.31 = 0.234$
  - $P(2 \mid \text{Yellow}) = 0.56/7.31 = 0.077$
  - $P(3 \mid \text{Yellow}) = 0.66/7.31 = 0.090$
  - $P(4 \mid \text{Yellow}) = 1.32/7.31 = 0.181$
  - $P(5 \mid \text{Yellow}) = 0.66/7.31 = 0.090$
  - $P(6 \mid \text{Yellow}) = 2.40/7.31 = 0.328$

Called	$P(\text{Yellow}\|X)$	$P(\text{Blue}\|X)$
6	.8	.2
4	.33	.67
5	.33	.67
1	.57	.43
2	.14	.86
3	.33	.67
4	.33	.67
5	.33	.67
2	.14	.86
2	.14	.86
1	.57	.43
4	.33	.67
3	.33	.67
4	.33	.67
6	.8	.2
2	.14	.86
1	.57	.43
6	.8	.2
	SUM=7.31	SUM=10.69

Expectation maximization

Step 3: Update $P(X|\text{Blue})$

- Total count for "Blue": 10.69
  - Total count for 1: 1.29
  - Total count for 2: 3.44
  - Total count for 3: 1.34
  - Total count for 4: 2.68
  - Total count for 5: 1.34
  - Total count for 6: 0.6
- Updated probability of Blue dice:
  - $P(1 \mid \text{Blue}) = 1.29/11.69 = 0.122$
  - $P(2 \mid \text{Blue}) = 0.56/11.69 = 0.322$
  - $P(3 \mid \text{Blue}) = 0.66/11.69 = 0.125$
  - $P(4 \mid \text{Blue}) = 1.32/11.69 = 0.250$
  - $P(5 \mid \text{Blue}) = 0.66/11.69 = 0.125$
  - $P(6 \mid \text{Blue}) = 2.40/11.69 = 0.0563$

Called	$P(\text{Yellow}\|X)$	$P(\text{Blue}\|X)$
6	.8	.2
4	.33	.67
5	.33	.67
1	.57	.43
2	.14	.86
3	.33	.67
4	.33	.67
5	.33	.67
2	.14	.86
2	.14	.86
1	.57	.43
4	.33	.67
3	.33	.67
4	.33	.67
6	.8	.2
2	.14	.86
1	.57	.43
6	.8	.2
	SUM=7.31	SUM=10.69

Expectation maximization

Step 4: Update $P(Z = \text{Blue})$ and $P(Z = \text{Yellow})$

- Total count for "Yellow": 7.31
- Total count for "Blue": 10.69
- Total instances: $7.31 + 10.69 = 18$
- We also normalize our estimate for the probability that the caller calls out Yellow or Blue
- $P(Z = \text{Yellow}) = 7.31/18 = 0.41$
- $P(Z = \text{Blue}) = 10.69/18 = 0.59$

Called	$P(\text{Yellow}\|X)$	$P(\text{Blue}\|X)$
6	.8	.2
4	.33	.67
5	.33	.67
1	.57	.43
2	.14	.86
3	.33	.67
4	.33	.67
5	.33	.67
2	.14	.86
2	.14	.86
1	.57	.43
4	.33	.67
3	.33	.67
4	.33	.67
6	.8	.2
2	.14	.86
1	.57	.43
6	.8	.2
	SUM=7.31	SUM=10.69

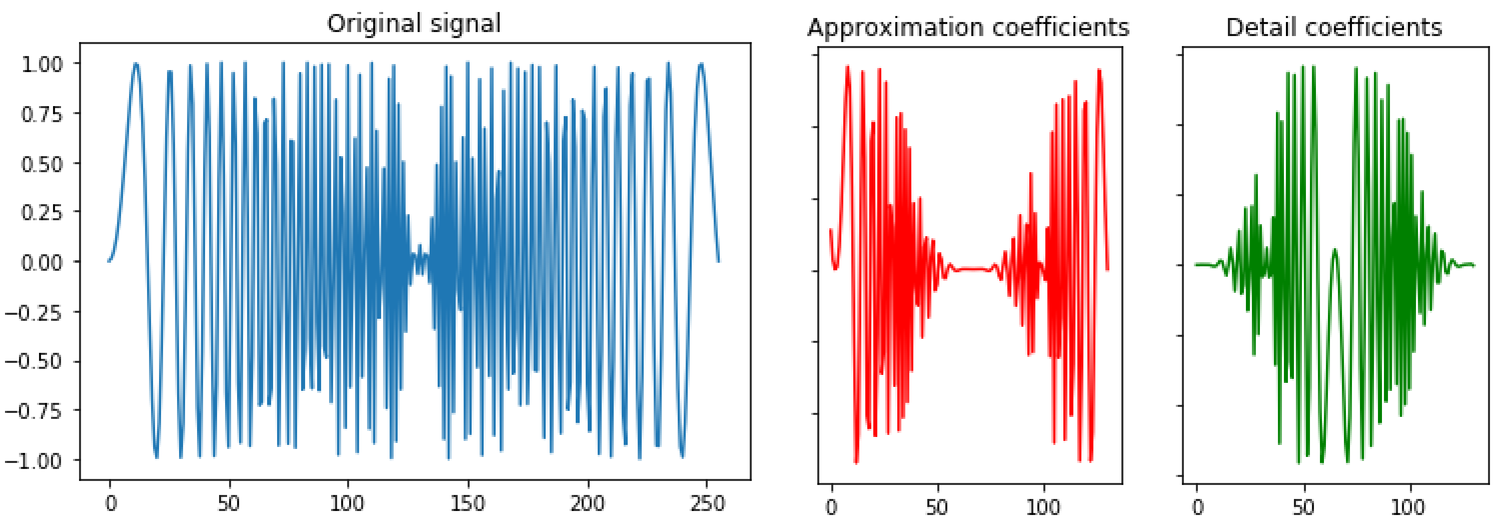
Expectation maximization: Summary

X	1	2	3	4	5	6
Initially, we randomly guess						
$P(X\|\text{Blue})$	0.3	0.3	0.1	0.1	0.1	0.1
$P(X\|\text{Yellow})$	0.4	0.05	0.05	0.05	0.05	0.4
Currently, we have updated results after one iteration of update						
$P(X\|\text{Blue})$	0.122	0.322	0.125	0.250	0.125	0.056
$P(X\|\text{Yellow})$	0.234	0.077	0.090	0.181	0.090	0.328

	Previous random guess	Updated results
$P(Z = \text{Blue})$	0.5	0.59
$P(Z = \text{Yellow})$	0.5	0.41

The *Expectation Maximization* (EM) algorithm will continue until the stopping criterion (e.g., # of iterations, etc).

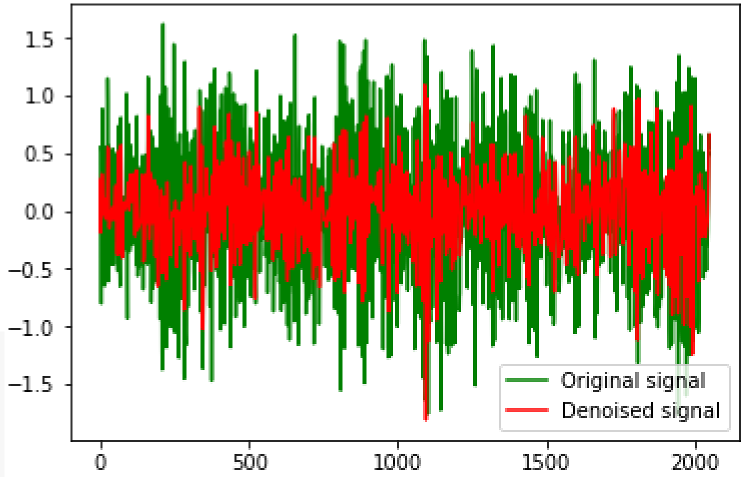
Perform wavelet decomposition on signal




```
# Step 1: Define input data and type of wavelet

# Data: Input data
data = chirp_signal
# waveletname: Type of wavelet
waveletname = 'db4'

# Step 2: Perform 1D wavelet decomposition
(data, coeff_d) = pywt.dwt(data, waveletname)
```

Workshop

Perform wavelet-based signal denoising

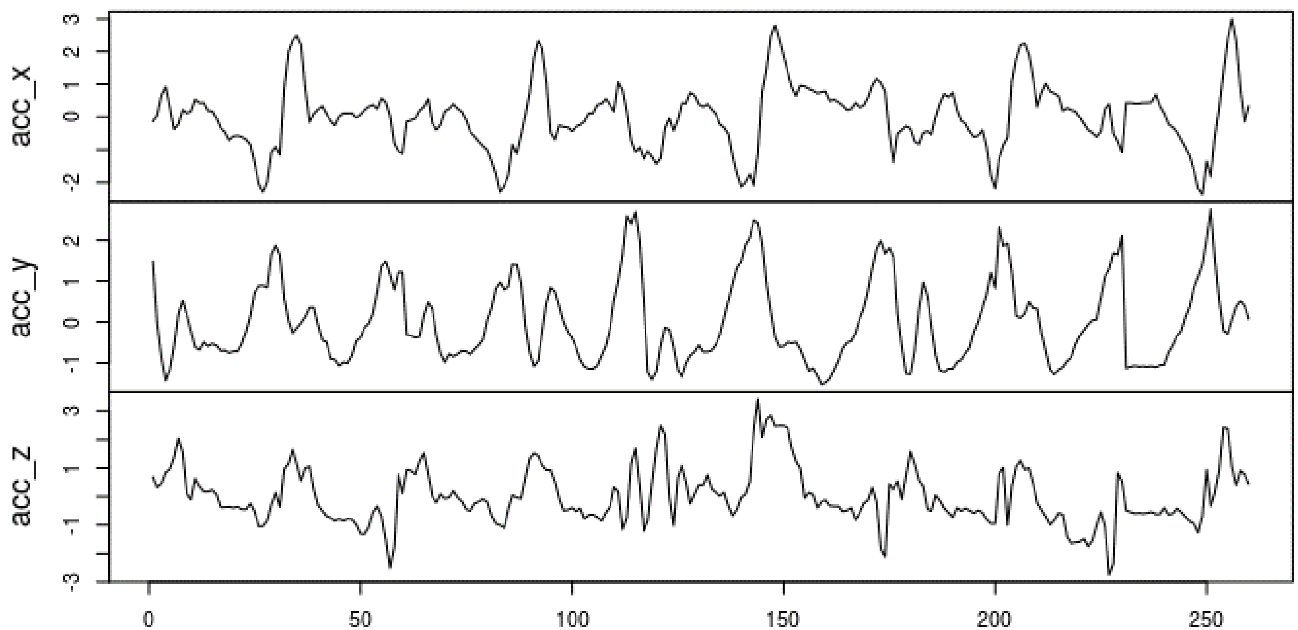


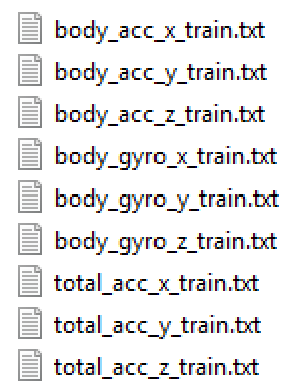

```
waveletname = 'db4'
waveletlevel = 2

# Step 1: Perform wavelet decomposition

coeffs_orig = pywt.wavedec(signal_orig, waveletname, level=waveletlevel)
coeffs_filter = coeffs_orig.copy()

# Step 2: Perform thresholding on the wavelet coefficients

# Set the threshold
threshold = 0.8

for i in range(1, len(coeffs_orig)):
    coeffs_filter[i] = pywt.threshold(coeffs_orig[i], threshold*max(coeffs_orig[i]))

# Step 3: Perform reconstruction on the filered coefficients

signal_denoised = pywt.waverec(coeffs_filter, waveletname)
```

Workshop

Human Activity Recognition Using Smartphones Data Set

- Extract statistical features from wavelet coefficients from wearable sensor data, and then perform classification for human activity classification
- Each person performed six activities (WALKING, WALKING_UPSTAIRS, WALKING_DOWNSTAIRS, SITTING, STANDING, LAYING) wearing a smartphone (Samsung Galaxy S II) on the waist. Using its embedded accelerometer and gyroscope, we captured 3-axial linear acceleration and 3-axial angular velocity at a constant rate of 50Hz.



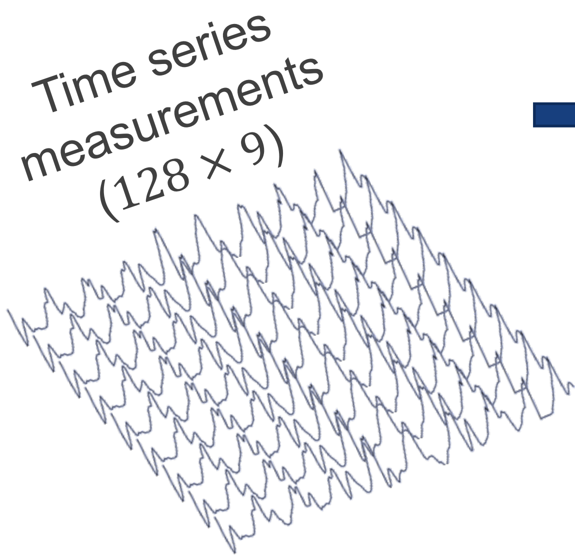


Reference: https://archive.ics.uci.edu/ml/datasets/Human+Activity+Recognition+Using+Smartphone

Workshop

Problem statement: Can we recognize human activity from smart phone sensory data? Note that: This is time series data classification, not time series data forecasting.

Model input	Model	Model output
Human activity smart phone data	To build a feature extraction method + statistical classification	Human activity (6 categories for this dataset)



Feature extraction + classification

Human activity categories (6)

# of measurements (2.56 seconds, 50 Hz)	128
# of sensors (X, Y, Z, for for total acceleration, body acceleration, body angular velocity)	9
# of activity	6

What we have learnt

- Bayesian signal classification
- Apply Bayesian signal classification on features extracted in wavelet domain

Thank you!

Dr TIAN Jing
Email: tianjing@nus.edu.sg