Advanced Concepts in Signal Processing Revision 2017

Dr S Tsaftaris

MSc in Digital Communications

Plan

- Part 1: Clarify points I received over email or in office hours
- Part 2: Go over a typical exam
  - How it looks; what you should do
  - Provide advice on the exam
- Part 3: Go over some problems from past exams.

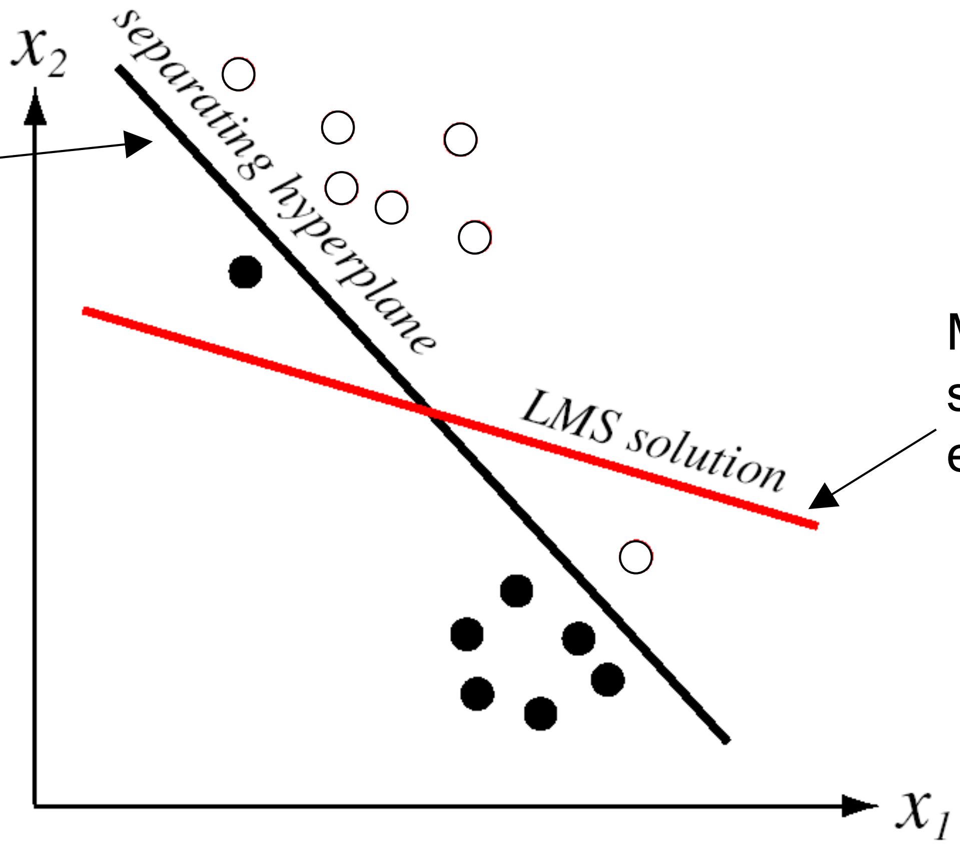
Part 1

MSE approach vs Perceptron

MSE and Separating Hyperplanes

Minimizing the squared error need not converge to a separating hyperplane solution, *even if one exists*.

Minimizes classification error



Minimizes squared error

MSE solution depends on the margins b_k used.

Learning in the Multicategory Case

Both MSE and Perceptron learning can be extended to multi-category systems.

For MSE, if we find the MSE solution to:

$$\mathbf{a}_i^t\mathbf{y} = 1 \quad \text{for all } \mathbf{y} \in Y_i \qquad \text{(i.e. } \mathbf{y} \text{ in category } i\text{)}$$

$$\mathbf{a}_i^t\mathbf{y} = 0 \quad \text{for all } \mathbf{y} \notin Y_i \qquad \text{(} \mathbf{y} \text{ not in category } i\text{)}$$

then it turns out that $\mathbf{a}_\mathrm{i}^\mathrm{T}\mathbf{y}$ is asymptotically (for a sufficient number of samples) an MSE approximation to:

$$P(\omega_i \,|\, \mathbf{x})$$

The probability of x being labelled ω_i. Therefore the decision rule:

$$\omega(\mathbf{y}) = \ \omega_i \text{ if } \mathbf{a}_i^T\mathbf{y} > \mathbf{a}_j^T\mathbf{y} \text{ for all } j \neq i$$

assigns the most probably category (more later).

This MSE solution is also a *linear machine*

Confused about the construction of B for multiclass MSE

The Pseudoinverse solution

We can now construct the multicategory pseudoinverse solution. Let **Y** be the set of samples partitioned into c sub-matrices:

$$\mathbf{Y} = \begin{bmatrix} \mathbf{Y}_1 \\ \vdots \\ \mathbf{Y}_c \end{bmatrix} \qquad \text{samples labelled } i \text{ are rows of } \mathbf{Y}_i$$

Let **A**=[**a**$_1$,…,**a**$_c$] be the matrix of weight vectors and define

$$\mathbf{B} = \begin{bmatrix} \mathbf{B}_1 \\ \vdots \\ \mathbf{B}_c \end{bmatrix}, \ \mathbf{B}_i \text{ is } 0, \text{except } i\text{th column is } 1.$$

The squared error: $\| (\mathbf{YA} - \mathbf{B}) \|_F^2 = \sum_i \| (\mathbf{Y}_i \mathbf{a}_i - \mathbf{b}_i) \|^2$

is minimized by: $\mathbf{A} = \mathbf{Y}^+ \mathbf{B}$

Questions related to Optimization

- Line search

- Steepest descent

- Conjugate gradient

What is a line search

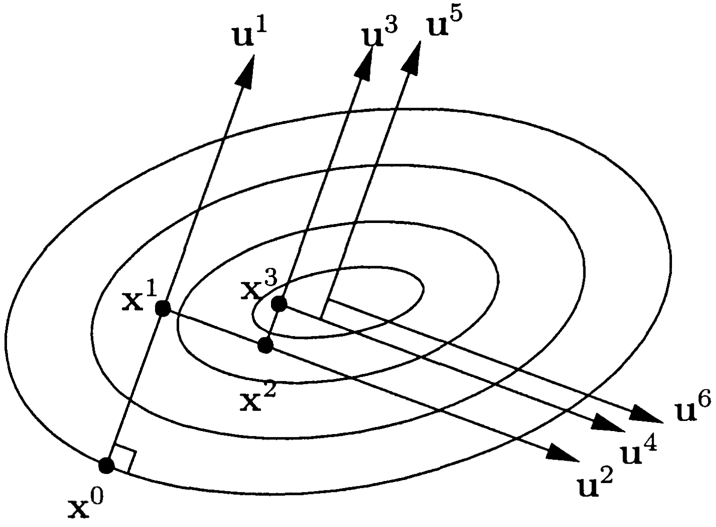
- Suppose I have a 2D function and I am a point w_o and a search direction d_1. How much should I move to get a new point w_1?
  - If I move a fixed step ➔ gradient descent
- What if I want to make an optimal step?
  - I am still on the line defined by the direction d_1
  - But I can find the minimum of the cost function moving along this line starting from w_o but keeping my “feet” always on the line the direction d_1 tells me.
  - This essentially finds $w_{i+1} = w_i + \lambda d_i$ changing only the lamda
  - We show many methods that can find this line minimum eg bracket methods

Then what is the difference between steepest descent and gradient descent

- Well simply gradient descent makes “fixed” steps ie the lambda is not optimized at each iteration
  - (Why is fixed in quotes? Because there are other methods that may modify the step size based on a formula of the number of elapsed iterations.)
- Steepest descent implies an optimum line minimum

So what is wrong with them…

- We always aim for best convergence so…
- Recall that if you do exact line minima then the next choice of direction is the negative gradient at that point.



- We saw that we then get orthogonal searches and bad zigzagging

How can I fix things?

Conjugate directions I

Construct directions that preserve the previous minimization work.

The idea of conjugate directions is to choose a direction $w = w_k + \lambda d_k$ such that:

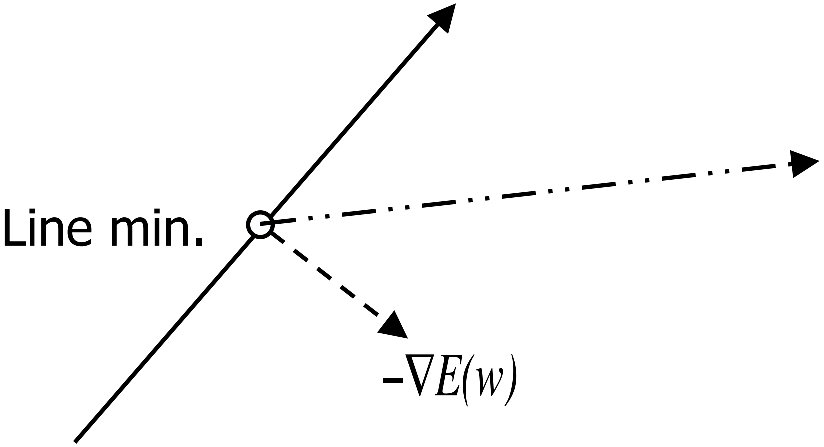
$$\nabla E(w_k + \lambda d_k)^T d_j = 0, \forall j < k$$

Which implies (with quadratic approx.)

$$(\nabla E(w_k) + \nabla^2 E(w_k)\lambda d_k)^T d_j = 0$$

➔

$$d_k^T \nabla^2 E(w_k) d_j = 0$$



Conjugate gradient algorithm I: nothing more than a line search of a quadratic function

Suppose we have a quadratic function:

$$E(w) = E_0 + b^T w + \frac{1}{2} w^T H w$$

Starting at w_i and searching in direction d_i the line minimum is:

$$w_{i+1} = w_i + \alpha_i d_i$$

denoting the gradient at w_i as $g_i = \nabla E(w_i) = b + H w_i$ we can solve for α_i

$$g_{i+1}^T d_i = (b + H(w_i + \alpha_i d_i))^T d_i = g_i^T d_i + \alpha_i d_i^T H d_i = 0$$

Hence:

$$\alpha_i = -\frac{g_i^T d_i}{d_i^T H d_i}$$

Conjugate gradient algorithm II

We now choose a d_{i+1} that is conjugate to d_i we will try a modified gradient:

$$d_{i+1} = -g_{i+1} + \beta_i d_i$$

for some β_i. Solving for conjugacy gives:

$$\left(-g_{i+1} + \beta_i d_i\right)^T H d_i = 0$$

Hence:

$$\beta_i = \frac{g_{i+1}^T H d_i}{d_i^T H d_i}$$

In fact this choice of direction is conjugate with all d_j, $j < i$.

Finally we can write:

$$\beta_i = \frac{g_{i+1}^T \left(\alpha_i H d_i\right)}{d_i^T \left(\alpha_i H d_i\right)} = \frac{g_{i+1}^T \left(g_{i+1} - g_i\right)}{d_i^T \left(g_{i+1} - g_i\right)}$$

since $g_{j+1} - g_j = H(w_{j+1} - w_j) = \alpha_j H d_j$ (no need to use H)

Is Maximum A Posteriori estimate same as Bayesian/Estimate Bayes?

Bayesian choice of *w*

Often we may wish to get a single value for **w** as an estimate from some data. From a Bayesian perspective this involves defining a loss function (c.f. Bayesian decision theory). 2 common choices are:

1. *Posterior Mean*:

$$\mathbf{w} = E\{\mathbf{w} \mid \chi\} = \int \mathbf{w}\, \mathrm{p}(\mathbf{w} \mid \chi)\, \mathrm{d}\,\mathbf{w}$$

this comes from a Mean Squared Error loss function.

2. ***Maximum a Posteriori* (MAP) estimates:**

$$\mathbf{w} = \underset{\mathbf{w}}{\mathrm{argmax}}\left[p(\mathbf{w} \mid \chi)\right]$$

closely related to ML estimation – not really *very* Bayesian (comes from a 0-1 loss function)

How does the k-means work in practice

- Let us consider a simple example:
  data {-10,-8, -1, 10,12,13}
  assume initial **centroids** -11, 11
  Write first iteration of k-means
  - assign points to centroids [Euclidean distance]
  - update centroids

data	-11	11	-6.33	11.66
-10	**1**	21	**3.67**	21.67
-8	**3**	19	**1.67**	19.67
-1	**10**	12	**5.33**	12.67
10	21	**1**	16.33	**1.67**
12	23	**1**	18.33	**0.33**
13	24	**4**	19.33	**1.33**

Follow-up questions

- Converged? Yes by updating the centroids again they don't change
- How can I write the final result as Y~ CW
- Y = [-10 -8 -1 10 12 13]
- C = [-6.33 11.66]
- W = [1 1 1 0 0 0]
  [0 0 0 1 1 1]

How does cross-validation helps identify bias/variance

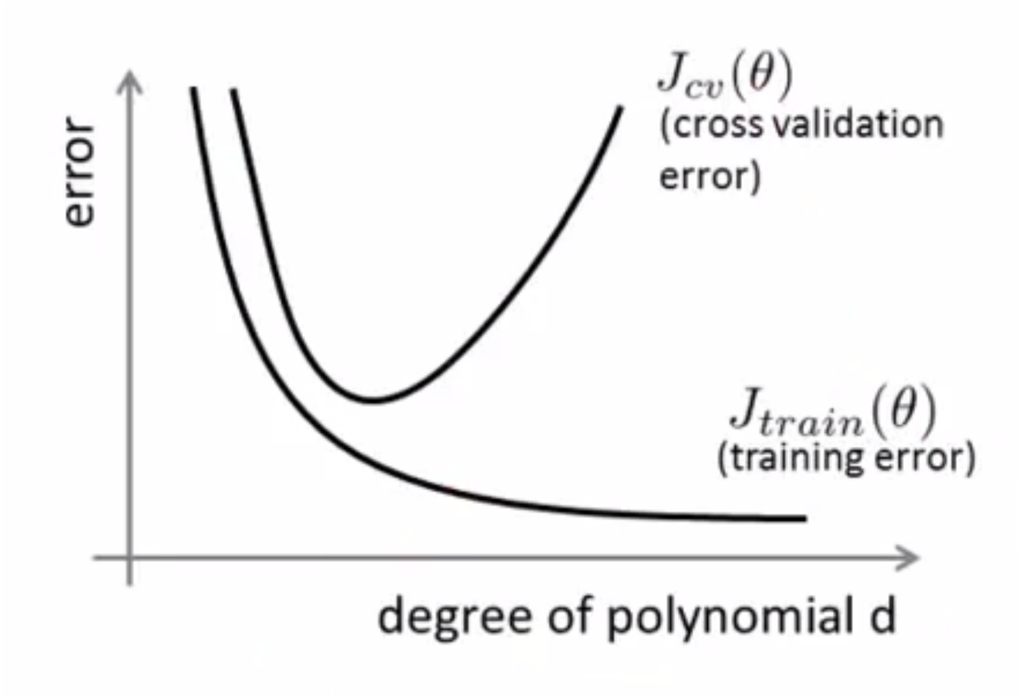
Overfitting in ML

- Too complex a model → high variance.
- Too simple a model → high bias.

Possible solution on diagnosis models and data (size)
Cross-validation: Break the data into **3 pieces**: *training*, *validation* and *testing* sets. Use the training data to learn the model parameters, but identify the best model order using the validation or "out-of-sample" data. Then report results on the training and testing sets.

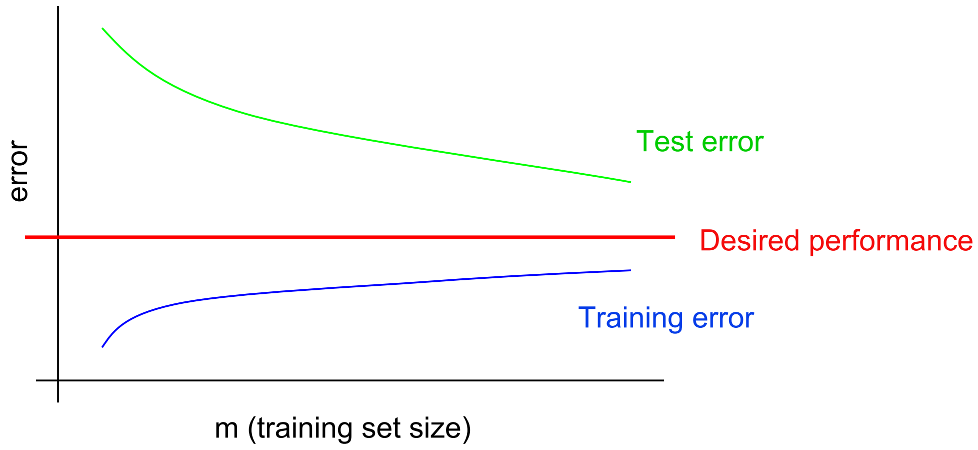
Diagnosing bias/variance: the practical picture

- For finding a good model size



Bias vs variance

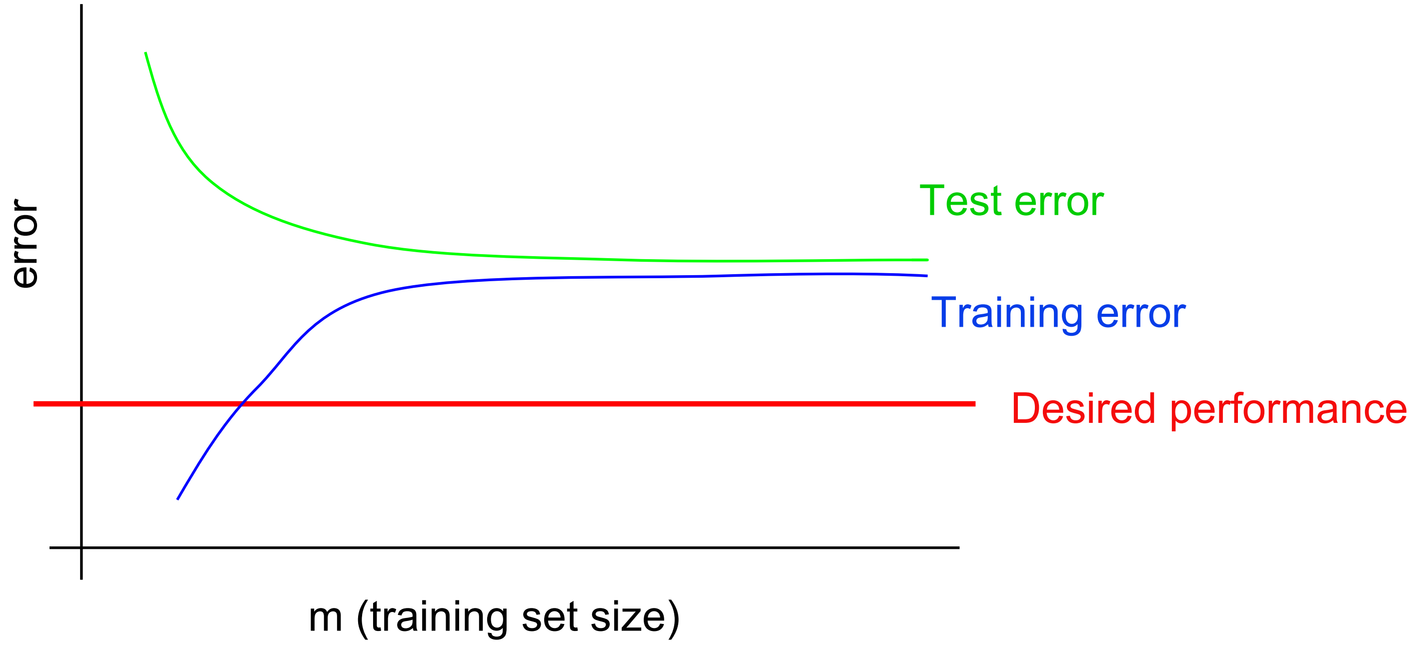
- Typical curve with high variance



- Test error decreases as m increases ➔ larger training set may help
- There is a gap between training & test error

Bias v variance

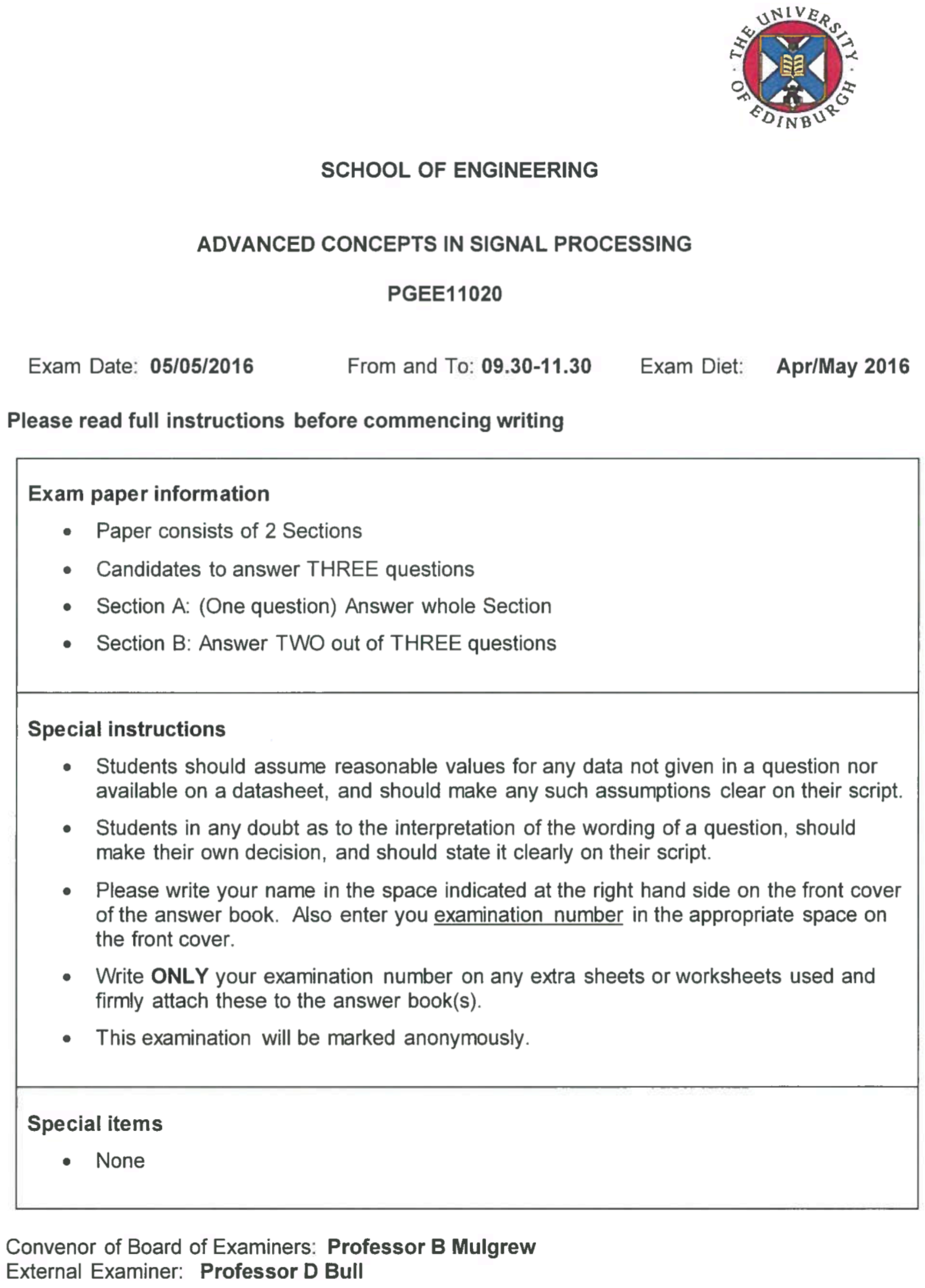
- Typical curve with high bias



- We are in trouble: training error is high too
- Small gap between the two errors

Part 2

The first page of an exam



THE UNIVERSITY OF EDINBURGH

SCHOOL OF ENGINEERING

ADVANCED CONCEPTS IN SIGNAL PROCESSING

PGEE11020

Exam Date: **05/05/2016** From and To: **09.30-11.30** Exam Diet: **Apr/May 2016**

Please read full instructions before commencing writing

Exam paper information

- Paper consists of 2 Sections
- Candidates to answer THREE questions
- Section A: (One question) Answer whole Section
- Section B: Answer TWO out of THREE questions

Special instructions

- Students should assume reasonable values for any data not given in a question nor available on a datasheet, and should make any such assumptions clear on their script.
- Students in any doubt as to the interpretation of the wording of a question, should make their own decision, and should state it clearly on their script.
- Please write your name in the space indicated at the right hand side on the front cover of the answer book. Also enter you examination number in the appropriate space on the front cover.
- Write **ONLY** your examination number on any extra sheets or worksheets used and firmly attach these to the answer book(s).
- This examination will be marked anonymously.

Special items

- None

Convenor of Board of Examiners: **Professor B Mulgrew**
External Examiner: **Professor D Bull**

in Signal

General advice on the exam

- Write your student code / name in the appropriate places and nowhere else
- Create legible solutions:
  - If we cant read it we cant grade it!
  - Stick to the boundaries of the page
  - Don’t shuffle between pages
- Read problem very well don’t try to “link” questions to what you have seen in the past & just playback memory
- Sometimes questions are wordy other times succinct. Usually a wordy question has a fast quick solution.
- Time yourself well

Part 3

Questions from Section A - 2016

SECTION A

Question A1

a) Explain what is meant by unsupervised learning and how it relates to mixture models. (**3**)

b) Explain the goal of Principal Component Analysis, identifying the underlying optimization problem. (**4**)

c) Define the concept of Independent Component Analysis indicating its key assumptions and ambiguities. (**4**)

d) Explain the concept of *Model Trust Region* methods for parameter optimization and indicate their relationship to the Newton and gradient descent methods. (**5**)

e) Describe the Levenberg-Marquardt update rule for minimising the sum of squared errors, indicating its key strengths. (**4**)

So a good answer for a) would be:

- Unsupervised learning is where we are only given input data but no output. [0.5]
- The task is then to learn a representation of the data often using latent variables. [0.5]
- Mixture models, model data using a mixture of hidden variables [0.5]

$$p(x) = \sum_{k=1}^{K} P(c_k) P(x \mid c_k) \qquad [1]$$

Notice the equation form. Is not the exact for GMM. The problem didn't ask me explicitly so I can write a more general form.

where $P(x \mid c_k)$ is component distribution.

It is also a form of unsupervised learning. [0.5]

- Observe the problem does not ask how to solve it; so no need to add more.

A good answer for b) would be

The goal of PCA is to approximate in a least-squares sense data x_i $x_i \in R^d$ using a reduced dimensional representation:

$$\tilde{x}_i \approx \sum_{i=1}^{K} z_i u_i + \bar{x}$$

$\bar{x}$ is data mean; u_i are the **principal vectors** and z_i **principal values**. [2]

The associated optimisation problem is a **LS optimisation** and can be solved via **eigenvalue decomposition** of the covariance matrix. Let

$$R_x =< x, x^T >= U \Lambda U^T$$

Λ diagonal eigenvalue matrix U eigenvectors The u_i above are the K vectors associated with the **largest eigenvalues.** [2]

A good answer for d)

To **stabilize** the Newton step we can restrict the region of search (and get good of both worlds [Newton and Gradient Descent]).

If M is symmetric +ve definite then $\Delta w = -M^{-1}\nabla E_0$ will point downhill.

Consider: $$\Delta w^{(k+1)} = -\left[\nabla^2 E(w^{(k)}) + \gamma I\right]^{-1} \nabla E(w^{(k)})$$

If $\gamma \to 0$ then $\mathrm{M} \to \nabla^2 E_0$ (Newton)

If $\gamma \to \infty$ then $\mathrm{M} \to (1/\gamma)I$ (Gradient descent, $\gamma = 1/\gamma$)

γ controls the size of the search region. We can choose γ adaptively.

e.g.

```
        If $E(w^{(k)}) < E(w^{(k-1)})$ then
                $\rightarrow \gamma = \gamma \div 10$
        else
                $\rightarrow w^{(k)} = w^{(k-1)}$ (ie reject update)
and             $\gamma = \gamma \times 10$ ie increase it
```

Question B2

A section B question from 2016

a) Write down an expression for a discriminant function, $z_k = g_k(\mathbf{x})$, generated by a three layer network in terms of its input-to-hidden weights, w_{ji}, hidden-to-output weights, $\tilde{w}_{kj}$, and input data $\mathbf{x}(n)$. Your solution should define any additional terms used. (**3**)

b) The network is going to be trained with a sum of squared error cost function:

$$J = \sum_n \sum_k (t_k(n) - z_k(n))^2$$

for training data $\mathbf{x}(n)$ where the target values, $\mathbf{t}(n)$, are set so that $t_k(n) = 1$ if $\mathbf{x}(n)$ is in class ω_k and $t_k(n) = 0$ otherwise. Explain in what way this provides an approximation to the optimal Bayes discriminant functions $P(\omega_k|\mathbf{x})$. (**3**)

c) Sketch the function, $f(u) = 1/(1 + e^{-u})$ and hence explain why it might be a good choice for the network activation function? (**2**)

d) Figure B2a shows the pth and $(p+1)$th layer of a deep neural network. Identify what is meant by the sensitivities, $\delta_i^{(p)}(n)$, of the network units at the pth layer and indicate their role in the Backpropagation of Errors algorithm. (**3**)

e) Show that the sensitivities for the pth layer can be written in terms of the sensitivities at the $(p+1)$th layer of the network. (**5**)

f) Derive the expression for the Backpropagation of Errors update rule for the pth layer-to-$(p+1)$th layer weights, $w_{ji}^{(p)}$. Hence show that this can be calculated using only the local pth layer variables and the local sensitivities. (**4**)

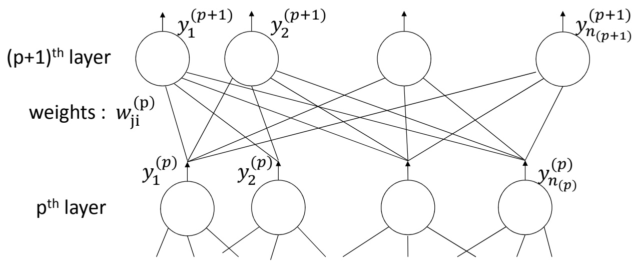


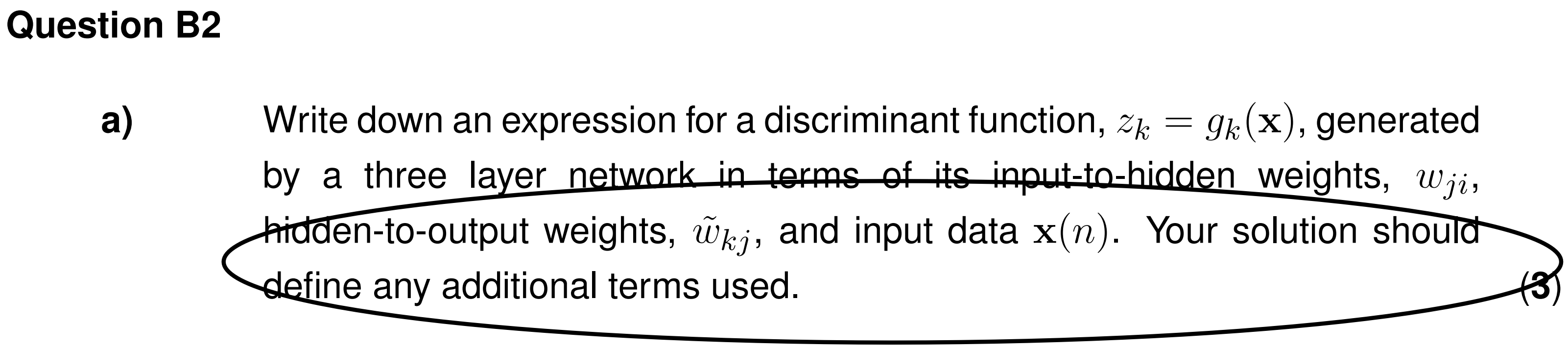
Figure B2a:

An incomplete answer for a)

$$z_k = g_k(x)$$

$$= f\left(\sum_{j=1}^{n_H} \tilde{w}_{kj}\, f\left(\sum_{i=1}^{n_I} w_{ji}\, x_i + w_{j0}\right) + \tilde{w}_{k0}\right)$$

- What is missing? Which will cost 1 point!



Question B2

a) Write down an expression for a discriminant function, $z_k = g_k(\mathbf{x})$, generated by a three layer network in terms of its input-to-hidden weights, w_{ji}, hidden-to-output weights, $\tilde{w}_{kj}$, and input data $\mathbf{x}(n)$. Your solution should define any additional terms used. (3)

- Define additional terms used!

A good answer for b)

b) if $t_k(n) = 1$ when $x(n) \in \omega_k$

$= 0$ otherwise

then the discriminant functions that minimise the sum of squared errors

in the large data limit

approximate the Posterior probabilities in the mean $\mathbb{E}\{$ square error sense

$$\mathbb{E}(J) = \mathbb{E}\left\{\left(g_k(x;w) - p(\omega_k|x)\right)^2\right\}$$

Is this a good answer for part e) ??

e)

$$\delta_i^{(p)}(n) = -\frac{\partial J}{\partial net_i^{(p)}(n)}$$

$$= -\frac{\partial J}{\partial y_i^{(p)}(n)} \frac{\partial y_i^{(p)}(n)}{\partial net_i^{(p)}(n)}$$

$$= -\sum_{j=1}^{n_{p+1}} \frac{\partial J}{\partial net_j^{(p+1)}(n)} \frac{\partial net_j^{(p+1)}(n)}{\partial y_i^{(p)}(n)} \frac{\partial y_i^{(p)}}{\partial net_i^{(p)}(n)}$$

$$= \sum_{j=1}^{n_{p+1}} \delta_j^{(p+1)}(n) \cancel{f(net_j^{(p+1)}(n))} w_{ji}^{(p)} \cdot f'(net_i^{(p)}(n))$$

It is

since $net_j^{(p)}(n) = \sum w_{ji}^{(p)} y_i^{(p)}(n)$ and $y_i^{(p)}(n) = f(net_i^{(p)}(n)$

Another B problem from 2014

a) Describe the key components of a Hidden Markov Model (HMM) based isolated word recognition system, identifying their specific roles. **(5)**

b) Let $\mathbf{A}^{(yes)}$ and $\mathbf{B}^{(yes)}$, below, define the state transition probabilities and emission probability matrix for a 4-state HMM for the word 'yes':

$$\mathbf{A}^{(yes)} = [\mathbf{a}_{ij}^{(yes)}] = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 \\ 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 1 \end{pmatrix} ; \mathbf{B}^{(yes)} = [\mathbf{b}_{jk}^{(yes)}] = \begin{pmatrix} 0.6 & 0.2 & 0.1 & 0.0 \\ 0.2 & 0.7 & 0.1 & 0.0 \\ 0.1 & 0.2 & 0.7 & 0.0 \\ 0.0 & 0.0 & 0.1 & 0.9 \end{pmatrix}$$

where the indices i, j and k range from 1 to 4.

Determine the possible distinct state sequences that could generate the observed sequence $\{v(1), v(2), v(3), v(4)\} = v_2 v_2 v_2 v_4$, assuming that the starting state at time $t = 0$ is ω_1. How many possible state sequences are there? **(4)**

c) Given a second HMM defined below for the word 'no', determine which of the two HMMs is more likely to have generated the observed sequence in part (c). Justify your answer.

$$\mathbf{A}^{(no)} = [\mathbf{a}_{ij}^{(no)}] = \begin{pmatrix} 0.5 & 0.5 & 0 \\ 0 & 0.5 & 0.5 \\ 0 & 0 & 1 \end{pmatrix} ; \mathbf{B}^{(no)} = [\mathbf{b}_{jk}^{(no)}] \begin{pmatrix} 0.2 & 0.3 & 0.5 & 0.0 \\ 0.5 & 0.3 & 0.2 & 0.0 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{pmatrix}$$

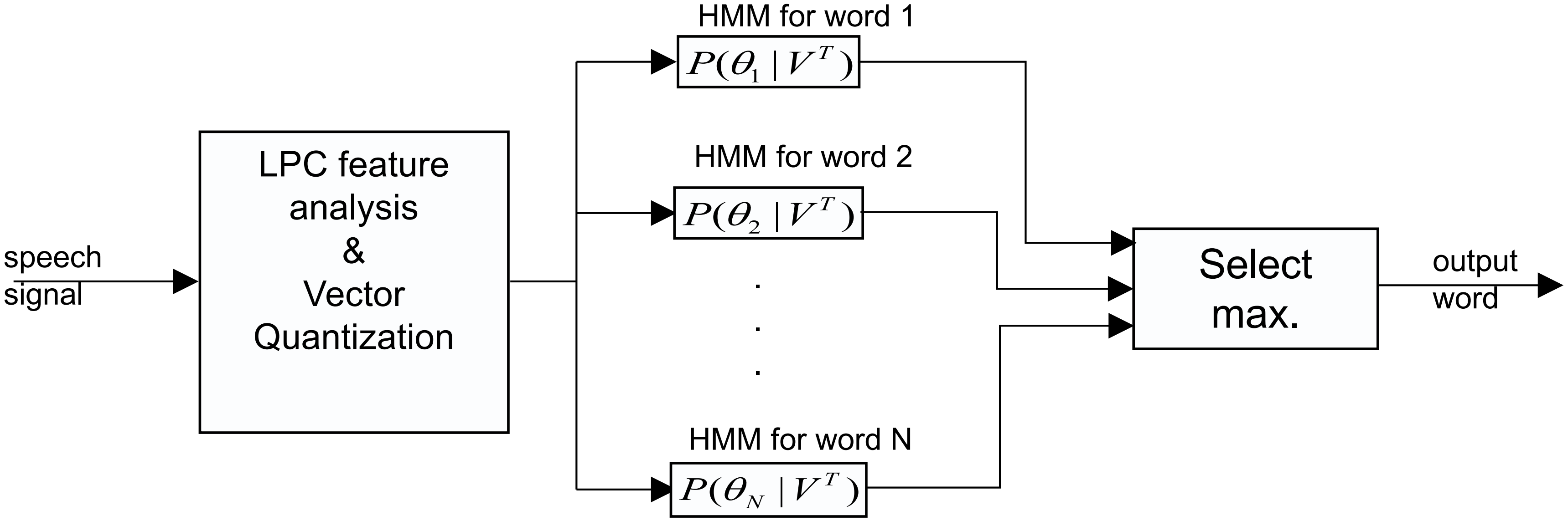
(7)

d) Describe the principles of HMM parameter selection. **(4)**

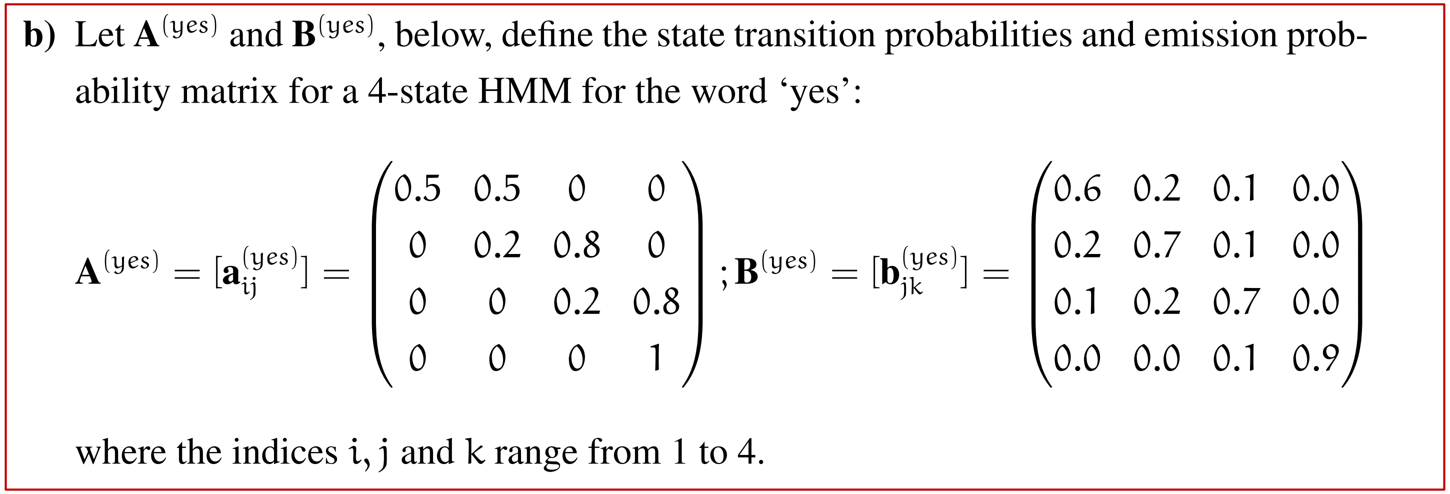
a) Describe the key components of a Hidden Markov Model (HMM) based isolated word recognition system, identifying their specific roles. (**5**)

In ASR the observed data is usually a measure of the short term spectral properties of the speech. There are two popular approaches:

1. *Continuous Density observations* – The finite states *ω(t)* are mapped into a continuous feature space using a MoG density model.
2. *VQ observations* – the continuous feature space is discretized into a finite symbol set using vector quantization.



An example of an isolated word HMM recognition system:

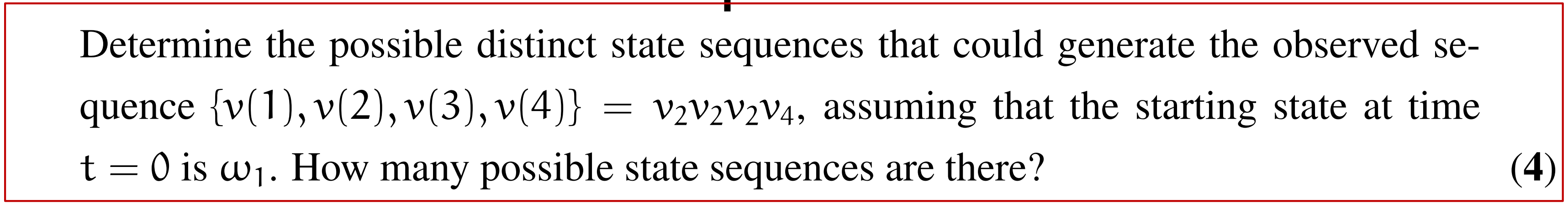


b) Let $\mathbf{A}^{(yes)}$ and $\mathbf{B}^{(yes)}$, below, define the state transition probabilities and emission probability matrix for a 4-state HMM for the word ‘yes’:

$$\mathbf{A}^{(yes)} = [\mathbf{a}_{ij}^{(yes)}] = \begin{pmatrix} 0.5 & 0.5 & 0 & 0 \\ 0 & 0.2 & 0.8 & 0 \\ 0 & 0 & 0.2 & 0.8 \\ 0 & 0 & 0 & 1 \end{pmatrix}; \mathbf{B}^{(yes)} = [\mathbf{b}_{jk}^{(yes)}] = \begin{pmatrix} 0.6 & 0.2 & 0.1 & 0.0 \\ 0.2 & 0.7 & 0.1 & 0.0 \\ 0.1 & 0.2 & 0.7 & 0.0 \\ 0.0 & 0.0 & 0.1 & 0.9 \end{pmatrix}$$

where the indices i, j and k range from 1 to 4.

- It helps to quickly draw state diagrams
- You will see that it is a left to right HMM!
- Observe the actual question.



Determine the possible distinct state sequences that could generate the observed sequence $\{v(1), v(2), v(3), v(4)\} = v_2 v_2 v_2 v_4$, assuming that the starting state at time $t = 0$ is ω_1. How many possible state sequences are there? **(4)**

- Most students will misread and they think this is a decoding problem! It is not exactly. In fact once you draw a trellis you will see that only few combinations of states can give that observed sequence

For part c)

- It gives us another HMM and asks us, if given the visible sequence which one of the two words is more likely.
- So this is an **evaluation** problem!
- Then we make decision according to which HMM maximizes the likelihood
- You can easily solve this either via the forward algorithm recursion in the algebraic steps or with a trellis diagram
- The final result you will see that yes is more probable.

For part d)

- This essentially asks you to describe parameter learning: how to find the state/emission matrices
- You assume that you have many utterances (features = visible sequences) of same word. → Labeled data
- You can use the principle of maximum likelihood.
- However the problem involves hidden variables.
- We can use something similar to Expectation Maximization iteratively estimating **a** and **b**
- We use the forward & backward algorithm to get estimates
- Then we average estimates across the data
- We repeat till convergence