ENGINEERING TRIPOS PART IIA

3F3

2009 Duration hours

SOLUTIONS

(TURN OVER

1. The Discrete Time Fourier Transform (DTFT) of a sequence $\{x_n\}$, $n = 0, \pm 1, \pm 2, \ldots, \pm\infty$ is given by

$$X(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} x_n e^{-jn\omega T}$$

where T is the sample period.

(a) Explain why the DTFT cannot in practice be evaluated on a digital computer and how this problem can be overcome. Show how the Discrete Fourier Transform (DFT)

$$X_p = \sum_{n=0}^{N-1} x_n e^{-j\frac{2\pi np}{N}}$$

can be derived directly from the DTFT expression by application of a rectangular window function sequence $\{w_n\}$,
$n = 0, \pm 1, \pm 2, \ldots, \pm\infty$

$$w_n = \begin{cases} 1, & \text{for } n = 0, 1, \ldots, N-1 \\ 0, & \text{otherwise} \end{cases}$$

and discretization of the frequency variable.

Answer

The DTFT cannot in practice be evaluated on a digital computer since

- the frequency variable ω is continuous
- the summation involves an infinite number of samples

The problems with computing DTFT on a digital computer can be overcome by:

- evaluating the DTFT at a finite collection of discrete frequencies
- performing the summation over a finite number of data points

The discrete set of frequencies is chosen arbitrary; however, since DTFT is periodic we generally choose a uniformly spaced grid of N frequencies covering the range $\omega T = 0 \rightarrow 2\pi$. Truncating the summation to just N data points is equivalent to an infinite summation but with x_n pre-multiplied by a rectangular window function

$$X_w(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} w_n x_n e^{-jn\omega T},$$

where

$$w_n = \begin{cases} 1, \text{ for } n = 0, 1, \ldots, N-1 \\ 0, \text{ otherwise} \end{cases}$$

and thus

$$\begin{aligned} X_w(e^{j\omega T}) &= \sum_{n=0}^{N-1} x_n e^{-jn\omega T} \\ X_p &= \sum_{n=0}^{N-1} x_n e^{-j\frac{2\pi np}{N}} \end{aligned}$$

(b) Show that the relationship between the DFT spectrum values X_p and the true DTFT spectrum $X(e^{j\omega T})$ may be expressed as a frequency domain convolution of the true spectrum and the spectrum $W(e^{j\theta})$ of the window function

$$\frac{1}{2\pi}\int_0^{2\pi} W(e^{j\theta})X(e^{j(\omega T-\theta)})d\theta$$

Determine an expression for the window spectrum $W(e^{j\theta})$.

Answer

The DTFT of the windowed signal $w_n x_n$, which is an equivalent to the DFT spectrum as was shown in (a) is given by

$$X_w(e^{j\omega T}) = \sum_{n=-\infty}^{\infty} w_n x_n e^{-jn\omega T},$$

If the spectrum of the window function is $W(e^{j\theta})$, w_n can be obtained using Inverse DTFT of $W(e^{j\theta})$:

$$w_n = \frac{1}{2\pi}\int_0^{2\pi} W(e^{j\theta})e^{jn\theta}d\theta$$

then

$$\begin{aligned} X_w(e^{j\omega T}) &= \sum_{n=-\infty}^{\infty}\left(\frac{1}{2\pi}\int_0^{2\pi} W(e^{j\theta})e^{jn\theta}d\theta\right)x_n e^{-jn\omega T} \\ &= \frac{1}{2\pi}\int_0^{2\pi} W(e^{j\theta})\sum_{n=-\infty}^{\infty} x_n e^{-jn(\omega T-\theta)}d\theta \\ &\quad \frac{1}{2\pi}\int_0^{2\pi} W(e^{j\theta})X(e^{j(\omega T-\theta)})d\theta \end{aligned}$$

The expression of the window spectrum can be computed using geometric progression as

$$W(e^{j\theta}) = \sum_{n=0}^{N-1} 1e^{-jn\theta} = \frac{1-e^{-j\theta N}}{1-e^{-j\theta}} = \frac{\sin(0.5\theta N)}{\sin(0.5\theta)} e^{-j0.5\theta(N-1)}$$

(c) Explain how the radix-2 Fast Fourier Transform (FFT) algorithm allows very efficient implementation of the DFT above when N is a power of 2. Your description should include: the number of stages in the radix-2 FFT algorithm and the number of "butterfly" computations required in relation to the transform size N; the "butterfly" structure and the computations required to compute one "butterfly" operation; bit reversal operations and in-place computation. Show that the total number of real operations (that is, real multiplications, additions or subtractions) required is approximately $5N \log_2 N$.

Answer

The radix-2 FFT algorithm has $M = log_2 N$ stages, each using $N/2$ butterflies. Each butterfly is defined by

$$X_p = A_p + W^p B_p \text{ and } X_{p+N/2} = A_p - W^p B_p.$$

where

$$A_p = \sum_{n=0}^{\frac{N}{2}-1} x_{2n} e^{-j\frac{2\pi np}{N/2}} \text{ and } B_p = \sum_{n=0}^{\frac{N}{2}-1} x_{2n+1} e^{-j\frac{2\pi np}{N/2}},$$

W^p are precomputed and stored.

Complex multiplication requires 4 real multiplications and 2 real additions; complex addition/subtraction requires 2 real additions, thus, one butterfly requires 10 real operations.

Thus, the radix-2 N-point FFT requires $10(N/2)log_2 N$ real operations compared to about $8N^2$ real operations for the DFT.

In order to get the correct output spectrum, the input data must be re-ordered by "bit reversal" of the binary expansion of their sample number.

Each step can be computed *in place* overwriting the same memory location.

2. (a) List the advantages and disadvantages of IIR filters in comparison with FIR filters.

Answer

- If the desired filter is highly selective (that is, its frequency response has small transition bandwidths or "steep sides"), then the impulse response will be long in the time domain. Examples include narrowband filters and lowpass /highpass /bandpass filters with steep cutoffs.
  For an FIR filter, a long impulse response means the filter is long (high order), so it requires many multiplications, additions and delays per sample.
  An IIR filter has active poles as well as zeros. Poles, acting as high-Q resonators, can provide highly selective frequency responses (hence long impulse responses) using much lower filter order than the equivalent FIR filter, hence much lower computational cost. (Although it is still true that a more selective response requires a higher order filter.)
- On the other hand, the closer to linear the phase is required to be, the higher the order of IIR filter that is needed. FIR filters can be linear phased though.
- The internal wordlengths in IIR filters need generally to be higher than those in FIR filters; this may increase the implementation cost (e.g in VLSI).
- An FIR filter is inherently stable, unlike an IIR filter. Hence an FIR implementation involving inaccurate (finite precision, or 'quantised') coefficients will be stable, whereas an IIR one might not. (However, it is desirable in either case to compute the actual frequency response of the filter, using the actual quantised values of the coefficients, to check the design.)
- IIR filter can suffer from limit cycles, deadbands, overflow, etc.

(b) Describe how the bilinear transform

$$s = \frac{1 - z^{-1}}{1 + z^{-1}}$$

can be used to convert analogue filters to digital filters. State the type of filters that can usefully be designed using this method, and any distortions that are introduced.

Answer

The bilinear transform produces a digital filter whose frequency response $H(e^{j\Omega})$ has the same characteristics as the frequency response of the analogue filter $H(j\omega)$, where $\Omega = \omega T$ (although its impulse response may then be quite different), i.e. the following important features are preserved:

- the $\omega \rightarrow \Omega$ mapping is monotonic
- $\omega = 0$ is mapped to $\Omega = 0$, and $\omega = \infty$ is mapped to $\Omega = \pi$ (half the sampling frequency). Thus, for example, a lowpass response that decays to zero at $\omega = \infty$ produces a lowpass digital filter response that decays to zero at $\Omega = \pi$
- the digital filter must be stable, thus, the bilinear transform maps the Left half s-plane onto the interior of the unit circle in the z-plane.

The steps of the bilinear transform method are as follows:

- warp the digital critical (e.g. bandedge or "corner") frequencies Ω_i, in other words compute the corresponding analogue critical frequencies $\omega_i = \tan(\Omega_i/2)$
- design an analogue filter which satisfies the resulting filter response specification
- apply the bilinear transform to the s-domain transfer function of the analogue filter to generate the required z-domain transfer function by replacing s with $\frac{1-z^{-1}}{1+z^{-1}}$

IIR filters can be designed with this method from analogue prototypes.

The main problem with bilinear transform is that it performs a nonlinear mapping of the phase leading to a distortion (or warping) of the digital frequency response. This effect is compensated by pre-warping the analogue filter before applying the linear transform.

(c) A bandpass filter is required in a channelised radio communications system with sampling rate 8kHz and 3dB cutoff frequencies of 2kHz and 3kHz. We are interested in designing such a filter from an analogue lowpass prototype given by

$$H(s) = \frac{1}{1+s}$$

which has a 3dB cutoff frequency of 1 rad/sec.

Using the lowpass to bandpass transformation

$$s' = \frac{s^2 + \omega_l \omega_u}{s(\omega_u - \omega_l)}$$

with lower cutoff at ω_l and upper cutoff at ω_u together with the bilinear transform, design the required digital filter.

Answer

The prewarped 3dB cutoff frequencies are

$$\begin{aligned} \omega_l &= \tan(\frac{1}{2}\frac{2000}{8000}2\pi) = 1 \\ \omega_u &= \tan(\frac{1}{2}\frac{3000}{8000}2\pi) = 2.4142 \end{aligned}$$

The lowpass to bandpass transformation is then

$$s' = \frac{s^2 + 2.4142}{1.4142s},$$

therefore, the band-pass filter is

$$\begin{aligned} H(s) &= \frac{1}{1+s'} \\ &= \frac{1}{1+\left(\frac{s^2+2.4142}{1.4142s}\right)} \\ &= \frac{1.4142s}{s^2+1.4142s+2.4142} \end{aligned}$$

Apply bilinear transform $\left(\frac{1-z^{-1}}{1+z^{-1}}\right)$. The corresponding digital filter is given by

$$\begin{aligned} H(z) &= \frac{1.4142\left(\frac{1-z^{-1}}{1+z^{-1}}\right)}{\left(\frac{1-z^{-1}}{1+z^{-1}}\right)^2 + 1.4142\left(\frac{1-z^{-1}}{1+z^{-1}}\right) + 2.4142} \\ &= \frac{1.4142\left(1-z^{-1}\right)\left(1+z^{-1}\right)}{\left(1-z^{-1}\right)^2 + 1.4142\left(1-z^{-1}\right)\left(1+z^{-1}\right) + 2.4142\left(1+z^{-1}\right)^2} \\ &= \frac{1.4142\left(1-z^{-2}\right)}{\left(1-2z^{-1}+z^{-2}\right) + 1.4142\left(1-z^{-2}\right) + 2.4142\left(1+2z^{-1}+z^{-2}\right)} \\ &= \frac{1.4142\left(1-z^{-2}\right)}{2*z^{-2} + 1.4142*2z^{-1} + 2*2.4142} \\ &= \frac{0.7071 - 0.7071z^{-2}}{z^{-2} + 1.4142z^{-1} + 2.4142} \\ &= \frac{0.2929 - 0.2929z^{-2}}{1 + 0.5858z^{-1} + 0.4142z^{-2}} \end{aligned}$$

(d) The filter is to be implemented in fixed precision digital hardware. Explain briefly the undesirable consequences of this implementation, and suggest the strategies for overcoming them.

Answer

If the result of any calculation in the filter exceeds its number range the overflow occurs. By default, a value slightly greater than the maximum representable positive number becomes a large negative number, and vice versa. This is called wraparound; the resulting error is huge. In IIR filters it can result in very large amplitude "overflow oscillations".

There are two strategies which can be used to avoid problems of overflow.

Scaling can be used to ensure that values can never (or hardly ever) overflow, and/or saturation arithmetic can be used to ensure that if overflow occurs its effects are greatly reduced.

3. The autoregressive moving-average (ARMA) model is a wide sense stationary process $\{x_n\}$ satisfying the equation:

$$\sum_{p=0}^{P} a_p x_{n-p} = \sum_{q=0}^{Q} b_q w_{n-q}$$

where $\{w_n\}$ is a white process with unity variance, and without loss of generality we assume $a_0 = b_0 = 1$.

(a) By interpreting $\{x_n\}$ as a result of filtering and assuming that the poles of the ARMA process all lie within the unit circle, derive the power spectrum of the above ARMA process.

Answer

Clearly the ARMA model is a pole-zero IIR filter based model with transfer function

$$H(z) = \frac{B(z)}{A(z)},$$

where

$$B(z) = \sum_{q=0}^{Q} b_q z^{-q}, \text{ and } A(z) = \sum_{p=0}^{P} a_p z^{-p}.$$

Thus, the frequency response is given by:

$$H(e^{j\omega T}) = \frac{B(e^{j\omega T})}{A(e^{j\omega T})},$$

and taking into account that $\{w_n\}$ is a white process with unity variance ($\sigma_w^2 = 1$), and $a_0 = b_0 = 1$, one obtains:

$$\begin{aligned} S_X(e^{j\omega T}) &= \sigma_w^2 \left|H(e^{j\omega T})\right|^2 \\ &= 1 \times \frac{\left|B(e^{j\omega T})\right|^2}{\left|A(e^{j\omega T})\right|^2} \\ &= \frac{\left|\sum_{q=0}^{Q} b_q e^{-j\omega qT}\right|^2}{\left|\sum_{p=0}^{P} a_p e^{-j\omega pT}\right|^2} \\ &= \frac{\left|1 + \sum_{q=1}^{Q} b_q e^{-j\omega qT}\right|^2}{\left|1 + \sum_{p=1}^{P} a_p e^{-j\omega pT}\right|^2} \end{aligned}$$

(b) Consider the following stationary random process as a particular case of the ARMA model presented above:

$$x_n = \sum_{q=0}^{Q} b_q w_{n-q}$$

where $\{w_n\}$ is a white process with unity variance, and without loss of generality we assume $b_0 = 1$. What type of process is this? Show that the autocorrelation function of this process is given by

$$r_{XX}[k] = \sum_{q=0}^{Q} b_q r_{XW}[k-q]$$

where $r_{XW}[k]$ is the crosscorrelation between x_n and w_n at lag k.

Answer

Since $a_p = 0$ for $p = 1, \ldots, P$, and $a_0 = 1$, this is a special case of the ARMA process called *moving average* (MA) process.

If the mean of the random process $\{x_n\}$ is defined as $E[x_n]$, and given that $\{x_n\}$ is a wide sense stationary process, the autocorre-

lation function is defined as:

$$\begin{aligned} r_{XX}[k] &= E[x_n x_{n+k}] \\ &= E\left[x_n \left(\sum_{q=0}^{Q} b_q w_{n+k-q}\right)\right] \\ &= \sum_{q=0}^{Q} b_q E\left[x_n w_{n+k-q}\right] \\ &= \sum_{q=0}^{Q} b_q r_{XW}[k-q] \end{aligned}$$

where $r_{XW}[k]$ is the crosscorrelation between x_n and w_n at lag k.

(c) Further show that the autocorrelation above can be expressed as:

$$r_{XX}[k] = \begin{cases} \sum_{l=0}^{Q-k} b_{l+k} b_l, & \text{for } k = 0, 1, \ldots, Q \\ 0, & \text{for } k > Q. \end{cases}$$

Answer

Given that

$$r_{XX}[k] = \sum_{q=0}^{Q} b_q r_{XW}[k-q],$$

let us consider

$$\begin{aligned} r_{XW}[k-q] &= E\left[x_n w_{n+k-q}\right] \\ &= E\left[\left(\sum_{i=0}^{Q} b_i w_{n-i}\right) w_{n+k-q}\right] \\ &= \sum_{i=0}^{Q} b_i E\left[w_{n-i} w_{n+k-q}\right] \end{aligned}$$

Taking into account that $\{w_n\}$ is stationary

$$\begin{aligned} r_{XW}[k-q] &= \sum_{i=0}^{Q} b_i E\left[w_{n-i} w_{n+k-q}\right] \\ &= \sum_{i=0}^{Q} b_i E\left[w_n w_{n+k-q+i}\right] \end{aligned}$$

and $\{w_n\}$ is a white noise process with unity variance

$$E\left[w_n w_{n+k-q+i}\right] = \begin{cases} 0, \text{ for } k-q+i \neq 0 \\ \sigma_w^2, \text{ otherwise} \end{cases}$$

where $\sigma_w^2 = 1$, thus,

$$E\left[w_n w_{n+k-q+i}\right] = \begin{cases} 0, \text{ for } i \neq q-k \\ 1, \text{ otherwise} \end{cases}$$

and ignoring all zero terms in the summation one obtains

$$\begin{aligned} r_{XW}[k-q] &= \sum_{i=0}^{Q} b_i E\left[w_n w_{n+k-q+i}\right] \\ &= b_{q-k} \end{aligned}$$

Thus,

- for $k \geq 0$ and $k \leq Q$

$$\begin{aligned} r_{XX}[k] &= \sum_{q=0}^{Q} b_q r_{XW}[k-q] \\ &= \sum_{q=0}^{Q} b_q b_{q-k} \end{aligned}$$

- for $k > Q$, the summation is zero since $b_q = 0$ for $q < 0$

(d) Given the following estimates of the autocorrelation function of a particular signal:

$$\begin{aligned} r_{XX}[1] &= -\frac{1}{2} \\ r_{XX}[k] &= 0 \text{ for } |k| > 1. \end{aligned}$$

determine the coefficient values for this particular model and roughly sketch the signal power spectrum.

Answer

First, $r_{XX}[k] = 0$ for $|k| > 1$ implies that $b_q = 0$ for all $q > 1$, i.e. it is MA(1) model and $Q = 1$.

Given that

$$r_{XX}[k] = \sum_{q=0}^{Q} b_q b_{q-k}$$

and taking into account that $b_q = 0$ for $q < 0$ and $b_0 = 1$ one obtains:

$$r_{XX}[1] = b_0 b_{-1} + b_1 b_0 = b_1 b_0 = b_1 = -0.5$$

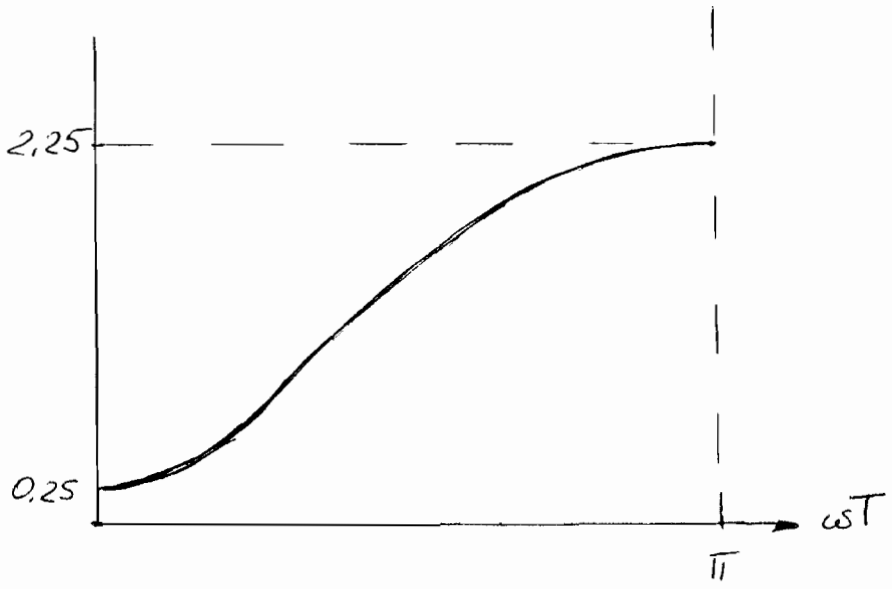
and the MA(1) model is

$$x_n = w_n - 0.5 w_{n-1}$$

and from (a) the power spectrum is given by

$$\begin{aligned} S_X(e^{j\omega T}) &= \frac{\left|1 + \sum_{q=1}^{Q} b_q e^{-j\omega qT}\right|^2}{\left|1 + \sum_{p=1}^{P} a_p e^{-j\omega pT}\right|^2} \\ &= \left|1 + \sum_{q=1}^{Q} b_q e^{-j\omega qT}\right|^2 \\ &= \left|1 - 0.5 e^{-j\omega T}\right|^2 \end{aligned}$$

When $\omega T = 0$ it takes value of $(1 - 0.5)^2 = 0.25$, at $\omega T = \pi$ it takes value of $(1 + 0.5)^2 = 2.25$



4. Consider the k-means clustering algorithm which seeks to minimise the cost function

$$C = \sum_{n=1}^{N} \sum_{k=1}^{K} s_{nk} \|x_n - m_k\|^2$$

where m_k is the mean (centre) of cluster k, x_n is data point n, $s_{nk} = 1$ signifies that data point n is assigned to cluster k, and there are N data points and K clusters.

(a) Given all the assignments $\{s_{nk}\}$, derive the value of m_k which minimises the cost C and give an interpretation in terms of the k-means algorithm.

Answer

Solve by taking derivatives and setting to zero.

$$\begin{aligned} \frac{\partial C}{\partial m_k} &= \sum_{n=1}^{N} s_{nk} \frac{\partial}{\partial m_k} (x_n - m_k)^{\mathsf{T}} (x_n - m_k) \\ &= \sum_{n=1}^{N} s_{nk} (-2x_n + 2m_k) = 0 \\ m_k &= \frac{\sum_{n=1}^{N} s_{nk} x_n}{\sum_{n=1}^{N} s_{nk}} \end{aligned}$$

This equation can be interpreted as follows: m_k is set to the mean of the data points assigned to cluster k.

(b) Give a probabilistic interpretation of k-means and describe how it can be generalised to unequal cluster sizes and non-spherical (elongated) clusters as shown in Fig. 1 below.

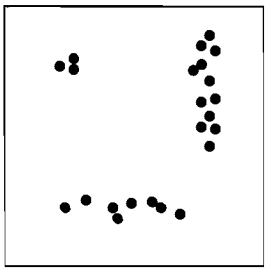


Figure 1:

Answer

K-means can be interpreted as an algorithm for fitting maximum likelihood parameters toa mixture of Gaussians where each Gaussian has spherically symmetric (i.e. isotropic) covariance matrix $\sigma^2 I$ and the Gaussians have equal proportions of data assigned to them $w_k = 1/K$ for all k (from lecture notes).

To generalise to unequal cluster sizes we allow w_k to vary, and to allow for elongated clusters we allow the covariance matrices for each Gaussian to vary and potentially be unequal.

(c) In many real-world applications, data points arrive sequentially and one wants to cluster them as they come in. Devise a sequential variant of the k-means algorithm which takes in one data point at a time and updates the means $\{m_1, \ldots, m_K\}$ sequentially without revisiting previous data points. Describe your sequential algorithm.

Answer

There are many possible answers, but here is one sequential variant of k-means:

- Assign the first K data points to the K clusters, and set $m_k = x_k$, and $n_k = 1$ (the number of points in cluster k).
- For each subsequent data point, x_n find the closest cluster centre, say m_k. Assign to this cluster and set:

$$m_k \leftarrow \frac{n_k}{n_k + 1} m_k + \frac{1}{n_k + 1} x_n$$

$$n_k \leftarrow n_k + 1$$

This algorithm has the property that m_k will always be the mean of all the data points assigned to it. One problem with this algorithm is that it is very senstitive to the first K points that arrive.