Lab 5 – Compute the Residual and Create AR Interpolation

Reminder on the Residual

Consider an AR process with a model order $P = 2$. We can write the prediction error or residual at sample k as

$$e_k = y_k - a_1 y_{k-1} - a_2 y_{k-2} \tag{3}$$

This is actually convolution of the signal y_k with an FIR filter that has taps $[1, -a_1, -a_2]$ i.e. $e_k = y_k \circledast h_k$ where h_k is an FIR filter with taps $h_0 = 1, h_1 = -a_1, h_2 = -a_2$. In Matlab use the `filter` to do this efficiently.

Lab 5a - Steps for Computing the Residual

- Write a function that estimates the residual for a vector of samples modelled with an AR model.
- Generate the residual for all the data samples, not just the "inner block". The data block input to the function is assumed to be normalised so all you are doing in the function is generating the residual using the prediction filter.
- If the model order is *p,* then there are *p* coefficients in the coeffs vector.
- The function must be able to be invoked as follows

```
[residual] = generateResidual(data, coeffs);
```

where the input vector of samples is "data", the "coeffs" vector contains "model order" coefficients in a vector starting from and ending with (model order = *p*).

- You can use the data contained in "referenceARsignal.mat" for checking your function. That data follows a 3rd order AR model.
- You are also given a script to invoke the function you will create.

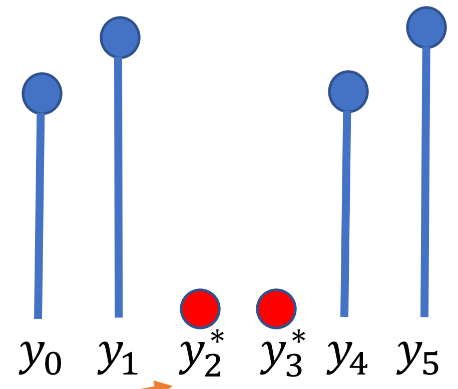
Reminder on the Interpolation

$$\mathbf{e} = \mathbf{A_k y_k} + \mathbf{A_u y_u}$$

Error split into known (y_k) and unknown (y_u)

We want to estimate $\boldsymbol{y_u}$ to minimise $\boldsymbol{e}^T\boldsymbol{e}$ so set $\boldsymbol{e} = \mathbf{0}$

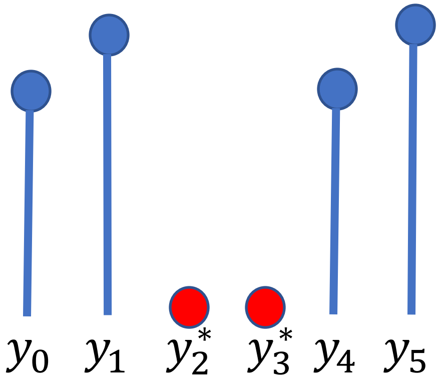
$$\begin{aligned} \mathbf{A_k y_k} + \mathbf{A_u y_u} &= 0 \\ \Rightarrow \mathbf{A_u' A_k y_k} + \mathbf{A_u' A_u y_u} &= 0 \\ \Rightarrow \mathbf{A_u' A_u y_u} &= -\mathbf{A_u' A_k y_k} \\ \Rightarrow \mathbf{y_u} &= -\left[\mathbf{A_u' A_u}\right]^{-1} \mathbf{A_u' A_k y_k} \end{aligned}$$



Remember that $\boldsymbol{y_u}$ in this case is just a two sample vector containing $[\boldsymbol{y_2^*}, \boldsymbol{y_3^*}]$

Interpolating with AR models

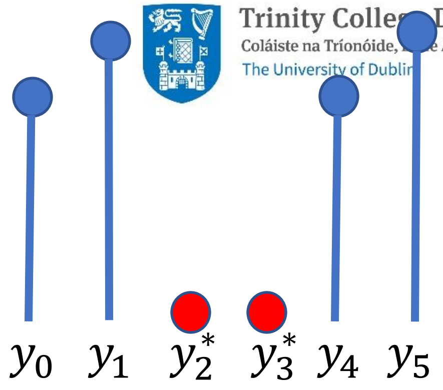
Prediction error equation at site j is $e_j = y_j - a_1 y_{j-1} - a_2 y_{j-2}$



$$\begin{bmatrix} e_2 \\ e_3 \\ e_4 \\ e_5 \end{bmatrix} = \begin{bmatrix} -a_2 & -a_1 & 1 & 0 & 0 & 0 \\ 0 & -a_2 & -a_1 & 1 & 0 & 0 \\ 0 & 0 & -a_2 & -a_1 & 1 & 0 \\ 0 & 0 & 0 & -a_2 & -a_1 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_2^* \\ y_3^* \\ y_4 \\ y_5 \end{bmatrix}$$

Write down the prediction error equations at sites 2,3,4,5
We want to estimate the values of the missing samples such that they minimise the prediction error power over the block

Interpolating with AR models



$$\begin{bmatrix} e_2 \\ e_3 \\ e_4 \\ e_5 \end{bmatrix} = \begin{bmatrix} -a_2 & -a_1 & 0 & 0 \\ 0 & -a_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & -a_1 & 1 \end{bmatrix} \begin{bmatrix} y_0 \\ y_1 \\ y_4 \\ y_5 \end{bmatrix} + \begin{bmatrix} 1 & 0 \\ -a_1 & 1 \\ -a_2 & -a_1 \\ 0 & -a_2 \end{bmatrix} \begin{bmatrix} y_2^* \\ y_3^* \end{bmatrix}$$

Rearrange the prediction equations like this ….

Lab 5b - Steps for interpolation

- Extract a block of the degraded data – plot the data suggested by the script
- Given a binary detection vector b (which is 1 where there is a click and a 0 otherwise) find the missing data from the block of degraded data – plot the data to see how many samples are missing
- Apply the AR-based interpolation to estimate the sample values at the locations of a 1 in the detection vector, i.e. write a function which is called like this

    ```
    [restored, Ak, Au, ik] = interpolateAR(datablock, detection, coeffs)
    ```

- You'll have to come up with a way to make the A_k, A_u and y_k matrices/vectors. *Hint: start by creating the matrix A first and then pull your sub-matrices (known - unknown) out of that!*
- Bonus: there is a much quicker and memory efficient way of doing this which does not require making up A but that is quite subtle