EE 263 Homework 3

Luis A. Perez

Memory of a linear dynamical, time-invariant system

Solution:

(a) Let us first consider how we might check if a valid impose response of fixed size M exists. The first thing to note is that the covolution operator given in the problem statent can actually be written as a linear system:

$$\bar{y} = Ah$$

where $\bar{y} \in \mathbb{R}^{T-M}, h \in \mathbb{R}^M$ and $A \in \mathbb{R}^{(T-M)\times M}$. Note that we remove the first $\{y_1, \cdots, y_M\}$. In fact, we can use the matrix A as defined below:

$$A = \begin{bmatrix} u_M & u_{M-1} & u_{M-2} & \cdots & u_1 \\ u_{M+1} & u_M & u_{M-1} & \cdots & u_2 \\ u_{M+2} & u_{M+1} & u_M & \cdots & u_2 \\ \vdots & \vdots & \vdots & \ddots & \\ u_{T-2} & u_{T-3} & u_{T-4} & \cdots & u_{T-M-1} \\ u_{T-1} & u_{T-2} & u_{T-3} & \cdots & u_{T-M} \end{bmatrix}$$

$$y_{-1} = \begin{bmatrix} y_{M+1} \\ \vdots \\ y_T \end{bmatrix}$$

$$h = \begin{bmatrix} h_1 \\ \vdots \\ h_M \end{bmatrix}$$

We can see by inspection above that Ah performs the needed convolutions betweens u and h to obtain $\bar{y}$, by the properties of matrix multiplication (eg, to obtain y_i, we compute the dot product of the $i-M$-th row of A with h, which is exactly what our convolution dictates, and for $i \leq M$, the convolution is not fully defined so we ignore them).

In the problem statement, we're given the fact that $T > 2M$, so this means that there will always be at least $T - M > 2M - M = M$ rows in A, meaning it will always be a tall and skinny matrix.

As such, we can use the pseudo inverse to find the closests solution for h, computing:

$$\bar{h} = (A^T A)^{-1} A^T \bar{y}$$

Finally, once we've computed this h, we can re-compute that $\bar{y}$ given our inputs forming A, and see if this equals our what we started with. In other words, we have that:

$$||A\bar{h} - \bar{y}|| \leq \epsilon \implies M \text{ is a valid value}$$

(ϵ is needed to deal with floating point imprecision)

The above gives us a way to check if M is sufficient. To finallize our method, we simply iterate over the possible values of M from $M = 1, \cdots \frac{T}{2} - 1$ in order until we find a valid value.

(b) Applying the process we described above, we find that $M = 7$ is the smallest value that works. We also have:

$$h = \begin{bmatrix} 0.63 \\ 0.27 \\ 0.02 \\ 0.37 \\ 0.96 \\ 0.95 \\ 0.46 \end{bmatrix}$$

Norm preserving implies orthonormal columns

Solution: We show that if $A \in \mathbb{R}^{m\times n}$ satifies $||Ax|| = ||x||$ for all $x \in \mathbb{R}^n$, then A has orthonormal columns.

This boils down to showing that $A^TA = I$, which can be broken into two parts. Let A_i be the i-th column of A. Then we need to show that $A_i^TA_i = 1$, and that $A_i^TA_j = 0$ for $i \neq j$. Let us consider the former first:

$$\begin{aligned} A_i^TA_i &= (Ae_i)^T(Ae_i) && \text{(Multiplying } A \text{ by } e_i \text{ extracts the } i\text{-th column)} \\ &= ||Ae_i||^2 && \text{(Definition of dot product treating } Ae_i \text{ as a vector)} \\ &= ||e_i||^2 && \text{(Multiplication by } A \text{ preserves the norm)} \\ &= 1 \end{aligned}$$

Let us now consider the latter case. We follow the hint:

$$\begin{aligned} ||A(e_i + e_j)||^2 &= ||e_i + e_j|| && (A \text{ preserves norm}) \\ &= 2 \end{aligned}$$

However, we also have that:

$$\begin{aligned} ||A(e_i + e_j)||^2 &= ||Ae_i + Ae_j||^2 && \text{(Linearity of } A) \\ &= ||Ae_i||^2 + ||Ae_j||^2 + 2(Ae_i)^T(Ae_j) && \text{(Definition of vector norm)} \\ &= 2 + 2(Ae_i)^T(Ae_j) && \text{(Using previous results)} \end{aligned}$$

Putting the two results together, we must have that:

$$2 + 2(Ae_i)^T(Ae_j) = 2 \implies 2(Ae_i)^T(Ae_j) = 0 \implies (Ae_i)^T(Ae_j) = 0 \implies A_i^TA_j = 0$$

As such, we've now shown that $A^TA = I$.

Sensor integrity monitor

Solution: We need to find $k \in \mathbb{R}^{k \times m}$ such that:

- $By = 0$ for any y which is consistent.
- $By \neq 0$ for any y which is inconsistent.

The above can be summarized as follow. Requirement (1) means that given $y \in \mathbf{Img}(A)$, we must have $By = 0$. This means that $\mathbf{Img}(A) \subseteq \mathbf{Ker}(B)$.

The second requirement specifies that for any $y \notin \mathbf{Img}(A)$, we must have $By \neq 0$. Phrase another way, this says that if $By = 0$, it must be in the $\mathbf{Img}(A)$. So we have $\mathbf{Ker}(B) \subseteq \mathbf{Img}(A)$.

As such, we're really just tring to construct a matrix B whose kernal is equal to the span of of the columns of A. This is actually a rather straight-forward process as long as we recall the following theorem, which holds for *any* matrix C:

$$\mathbf{Ker}(C^T) = \mathbf{Img}(C)^\perp$$

Thinking about it step-by-step, if the Kernel of B is equal to the Image of A, it must mean that the Image of B is equal to the complement of the Image of A. Using the formula presented abouve, the completement of the Image of A is the null-space of of A^T.

So simply put, we just need to (1) find the $M - N$ vectors spanning the nullspace of A^T and take these to be the rows of B. This will guarantee that the Kernel of B will equal the image of A.

Doing exactly the process described above gives us $B \in \mathbb{R}^{2 \times 5}$ as follows:

$$B = \begin{bmatrix} 0 & -5.47259193 & 2 & 8.47259193 & 1 \\ 0 & 6.97831636 & 6.39848918 & 2.61941742 & 3.19924459 \end{bmatrix}$$

Sensor integrity monitor

Solution:

(a) TODO

Solving linear equations via QR Factorization

Solution: The problem statement essentially boils down to solving the system $Rx = z$ where $R \in \mathbb{R}^{n\times n}$ is upper triangular and non-singular (eg, $R_{ii} \neq 0, \forall i$) and $z, x \in \mathbb{R}^n$.

The algoritm proceeds as follows:

1. First, let us find x_n. Let $R_i \in \mathbb{R}^{1\times n}$ be the i-th row of R as a row vector. We have:

$$\begin{aligned} z_n &= R_n x \\ &= \sum_{i=1}^{n} R_{ni} x_i \\ &= R_{nn} x_n && \text{(Since } R \text{ is upper triangular, } R_{ni} = 0 \text{ for } i < n) \\ \implies x_n &= \frac{z_n}{R_{nn}} && \text{(Note this is valid since } R_{nn} \neq 0) \end{aligned}$$

2. Using a similar process, we can now find x_{n-1}. We have:

$$\begin{aligned} z_{n-1} &= R_{n-1} x \\ &= \sum_{i=1}^{n} R_{n-1,i} x_i \\ &= R_{n-1,n-1} x_{n-1} + R_{n-1,n} x_n \\ & && \text{(Since } R \text{ is upper triangular, } R_{n-1,i} = 0 \text{ for } i < n-1) \\ \implies x_{n-1} &= \frac{z_n - R_{n-1,n} x_n}{R_{n-1,n}} && \text{(Note this is valid since } R_{n-1,n} \neq 0) \end{aligned}$$

   While the equation above looks more complicated, we note that all of the variables in the RHS are known (specifically, we found the value of x_n previously).

3. We simply continue the process from above. Let us assume that we've found the values x_k for all $k > i$. Then to compute x_i (the next value), we have:

$$\begin{aligned} z_i &= R_i x \\ &= \sum_{j=1}^{n} R_{ij} x_j \\ &= \sum_{j=i}^{n} R_{ij} x_j && \text{(Since } R \text{ is upper triangular, } R_{ij} = 0 \text{ for } j < i) \\ \implies x_i &= \frac{z_i - \sum_{j=i+1}^{n} R_{ij} x_j}{R_{ii}} && \text{(Note this is valid since } R_{ii} \neq 0) \end{aligned}$$

   We know $z_i, R_{ij}, \forall j$, and by our assumption, we know $x_k, \forall k > i$, so we know all the variables in the LHS. As such, it's relatively straight-forwrad to compute x_i.

As such, we can see that the $x_n, x_{n-1}, \cdots, x_1$ produced by our algorithm are valid solutions to the system $z = Rx$. Furthermore, the algorithm cannot fail since R is non-singular (all diagonal entries are non-zero).

Quadratic extrapolation of a time series, using least-squares fit

(a) In the problem statement, we are given that $\hat{z}(t+1) = f(t+1)$ where $f(\tau) = a_2\tau^2 + a_1\tau + a_0$ is the quadratic function which minimizes:

$$J = \sum_{\tau=t-9}^{t} (z(\tau) - f(\tau))^2$$

We can rewrite the above in matrix form as follows:

$$J = ||Fa - z||^2$$

where

$$z = \begin{bmatrix} z(t) \\ z(t-1) \\ \vdots \\ z(t-9) \end{bmatrix}$$

$$a = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix}$$

$$F = \begin{bmatrix} 1 & t & t^2 \\ 1 & t-1 & (t-1)^2 \\ \vdots & \vdots & \vdots \\ 1 & t-9 & (t-9)^2 \end{bmatrix}$$

If we want to solve for a, we can immediately arrive at the solution (based on what we've covered in lecture):

$$a = (F^T F)^{-1} F^T z$$

With the above, we can write $\hat{z}(t+1)$ as follows:

$$\begin{aligned} \hat{z}(t+1) &= f(t+1) \\ &= a_1 + a_2(t+1) + a_3(t+1)^2 \\ &= \begin{bmatrix} 1 & t+1 & (t+1) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} \\ &= \begin{bmatrix} 1 & t+1 & (t+1) \end{bmatrix} (F^T F)^{-1} F^T z \end{aligned}$$

As such, we can see that for a given value of t, we have:

$$\hat{z}(t+1) = \begin{bmatrix} 1 & t+1 & (t+1)^2 \end{bmatrix} (F^T F)^{-1} F^T \begin{bmatrix} z(t) \\ z(t-1) \\ \vdots \\ z(t-9) \end{bmatrix}$$

which implies that we should have:

$$c = \begin{bmatrix} 1 & t+1 & (t+1)^2 \end{bmatrix} (F^T F)^{-1} F^T$$

which at first glance appears to depend on the value of t. However, using code to solve the system symbolically, we see that the result is:

$$c$$