QBUS1040: Foundations of Business Analytics
Homework 4

Semester 2, 2019

This homework consists of eight problems that require you to submit a written response. You need to print it and write your answers directly in this paper. You should use scratch paper (which you will not turn in) to do your rough work. You should submit a scanned copy of your written solution as a 7-page PDF file via Canvas.

You must not submit photos of your written solution as it is difficult for the marking system to recognise your submission. Also, please do not use a tablet to write your homework as it is very likely that your submission will not be processed correctly through the marking system. You must use a conventional scanner. You should double check your PDF before you submit the file. The file size of your PDF document should not exceed 128MB, or it will not be accepted by the submission system.

The homework is due by 5pm on Monday, the 4th of November. **Late homework will not be accepted. Violation of the submission instructions may incur a 30% penalty.**

For all problems where you are asked for a free-form answer, it must be written in the box below the problem. We won't read anything outside the boxes.

All problems have equal weight. Some are easy. Others, not so much.

Tutorial time: Friday 4pm

Tutor's name: Mr Kam Fung (Henry) Cheung

Your SID: 4 8 0 0 4 8 6 8 1

(For QBUS1040 staff only)

Question:	1	2	3	4	5	6	7	8	Total
Points:	10	10	10	10	10	10	10	10	80
Score:									

2. (10 points) *Tall-wide product* Suppose A is an $n \times p$ matrix and B is a $p \times n$ matrix, so $C = AB$ makes sense. Explain why C cannot be invertible if A is tall and B is wide, i.e., if $p < n$. *Hint.* First argue that the columns of B must be linearly dependent.

- If B is wide, that is $n > p$, then based on the independence-dimension inequality, the columns of B must be linearly dependent.
- By the same argument, we can also conclude that the rows of A are linearly dependent.

3. (10 points) *Simultaneous left inverse* The two matrices

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 1 \\ 2 & 1 \\ 2 & 2 \end{bmatrix} \quad B = \begin{bmatrix} 3 & 2 \\ 1 & 0 \\ 2 & 1 \\ 1 & 3 \end{bmatrix}$$

and both left-invertible, and have multiple left inverses. Do they have a common left inverse? Explain how to find a 2×4 matrix C that satisfies $CA = CB = I$, or determine that no such matrix exists. (You can use numerical computing to find C.) *Hint.* Set up a set of linear equations for the entries of C. *Remark.* There is nothing special about the particular entries of the two matrices A and B.

Let C be a 2×4 matrix with its elements $\begin{bmatrix} c_1 & c_2 & c_3 & c_4 \\ c_5 & c_6 & c_7 & c_8 \end{bmatrix}$

$$\Rightarrow CA = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 \\ c_5 & c_6 & c_7 & c_8 \end{bmatrix} \begin{bmatrix} 1 & 2 \\ 3 & 1 \\ 2 & 1 \\ 2 & 2 \end{bmatrix} = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Rightarrow \begin{cases} c_1 + 3c_2 + 2c_3 + 2c_4 = 1 & (1) \\ 2c_1 + c_2 + c_3 + 2c_4 = 0 & (2) \\ c_5 + 3c_6 + 2c_7 + 2c_8 = 0 & (3) \\ 2c_5 + c_6 + c_7 + 2c_8 = 1 & (4) \end{cases} \qquad \begin{cases} 3c_1 + c_2 + 2c_3 + c_4 = 1 & (5) \\ 2c_1 + c_3 + 3c_4 = 0 & (6) \\ 3c_5 + c_6 + 2c_7 + c_8 = 0 & (7) \\ 2c_5 + c_7 + 3c_8 = 1 & (8) \end{cases}$$

$$\text{while } CB = \begin{bmatrix} c_1 & c_2 & c_3 & c_4 \\ c_5 & c_6 & c_7 & c_8 \end{bmatrix} \begin{bmatrix} 3 & 2 \\ 1 & 0 \\ 2 & 1 \\ 1 & 3 \end{bmatrix} = I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$\Rightarrow$ Solve equations (1), (2), (5), (6) to find c_1, c_2, c_3, and c_4.
And equations (3), (4), (7), (8) to find c_5, c_6, c_7, c_8. $\Rightarrow C = \frac{1}{11}\begin{bmatrix} -3 & -2 & 12 & -2 \\ 6 & 4 & -13 & 4 \end{bmatrix}$

4. (10 points) *Inverse of a block upper triangular matrix* Let B and D be invertible matrices of sizes $m \times m$ and $n \times n$, respectively, and let C be any $m \times n$ matrix. Find the inverse of

$$A = \begin{bmatrix} B & C \\ 0 & D \end{bmatrix}$$

in terms of B^{-1}, C and D^{-1}. (The matrix A is called *block upper triangular.*)

Hints. First get an idea of what the solution should look like by considering the case when B, C, and D are scalars. For the matrix case, your goal is to find matrices W, X, Y, Z (in terms of B^{-1}, C, and D^{-1}) that satisfy

$$A \begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = I$$

Use block matrix multiplication to express this as a set of four matrix equations that you can then solve. The method you will find is sometimes called *block back substitution.*

Assume there exists a block matrix $\begin{bmatrix} W & X \\ Y & Z \end{bmatrix}$ such that

$$A \cdot \begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = I$$

$$\Rightarrow \begin{bmatrix} B & C \\ 0 & D \end{bmatrix}\begin{bmatrix} W & X \\ Y & Z \end{bmatrix} = I$$

$$\Rightarrow \begin{bmatrix} BW + CY & BX + CZ \\ 0W + DY & 0X + DZ \end{bmatrix} = I$$

$$\Rightarrow \begin{cases} BW + CY = I \\ BX + CZ = \cancel{I}\,0 \\ DY = 0 \\ DZ = I \end{cases} \Leftrightarrow \begin{cases} BW = I \\ BX + C \cdot D^{-1} = 0 \\ Y = 0 \\ Z = D^{-1} \text{ (since } D \text{ is invertible)} \end{cases}$$

$$\Rightarrow \begin{cases} W = B^{-1} \\ X = -B^{-1}CD^{-1} \\ Y = 0 \\ Z = D^{-1} \end{cases}$$

So the inverse of A is $\begin{bmatrix} B^{-1} & -B^{-1}CD^{-1} \\ 0 & D^{-1} \end{bmatrix}$.

5. (10 points) *Least angle property of least squares* Suppose the $m \times n$ matrix A has linearly independent columns, and b is an m-vector. Let $\hat{x} = A^{daggar}b$ denote the least squares approximate solution of $Ax = b$.

   (a) Show that for any n-vector x, $(Ax)^T b = (Ax)^T(A\hat{x})$, i.e., the inner product of Ax and b is the same as the inner product of Ax and $A\hat{x}$.
   Hint. Use $(Ax)^T b = x^T(A^T b)$ and $(A^T A)\hat{x} = A^T b$.

We have the LHS $= (Ax)^T b$

$= x^T (A^T b)$

$= x^T [(A^T A)\hat{x}]$ (1)

while the RHS $= (Ax)^T (A\hat{x})$

$= x^T A^T A \hat{x}$ (2)

since (1) = (2), we can conclude that $(Ax)^T b = (Ax)^T (A\hat{x})$, that is the inner product of Ax and b is the same as the inner product of Ax and $A\hat{x}$.

   (b) Show that when $A\hat{x}$ and b are both nonzero, we have $\dfrac{(A\hat{x})^T b}{\|A\hat{x}\|\|b\|} = \dfrac{\|A\hat{x}\|}{\|b\|}$. The left-hand side is the cosine of the angle between $A\hat{x}$ and b. *Hint.* Apply part (a) with $x = \hat{x}$.

We have $(A\hat{x})^T b = \|A\hat{x}\| . \|b\| . \cos(A\hat{x}, b)$ (by definition)

$\Rightarrow \cos(A\hat{x}, b) = \dfrac{(A\hat{x})^T b}{\|A\hat{x}\| . \|b\|}$ (1)

We also have $(Ax)^T(A\hat{x}) = \|Ax\|. \|A\hat{x}\| . \cos(Ax, A\hat{x})$

and since $(A\hat{x})^T b = (Ax)^T(A\hat{x})$ from part a

$\Rightarrow \cancel{\|A\hat{x}\|} . \|b\| . \cos(A\hat{x}, b) = \|Ax\| . \cancel{\|A\hat{x}\|} . \cos(Ax, A\hat{x})$

$\cos(A\hat{x}, b) = \dfrac{\|Ax\| . \cos(Ax, A\hat{x})}{\|b\|}$

When $x = \hat{x}$, we have $\cos(A\hat{x}, b) = \dfrac{\|A\hat{x}\| . \overbrace{\cos(A\hat{x}, A\hat{x})}^{=1}}{\|b\|}$ (2)

From (1), (2) we conclude that $\dfrac{(A\hat{x})^T b}{\|A\hat{x}\| . \|b\|} = \dfrac{\|A\hat{x}\|}{\|b\|}$

(c) Least angle property of least squares. The choice $x = \hat{x}$ minimizes the distance between Ax and b. Show that $x = \hat{x}$ also minimizes the angle between Ax and b. (You can assume that Ax and b are nonzero.) *Remark.* For any positive scalar α, $x = \alpha\hat{x}$ also minimizes the angle between Ax and b.

Let θ be the angle between Ax and b

Since $\hat{x}$ is the least squares approximate solution of $A^{\text{daggar}} b$, we can write $b = A\hat{x}$

When $x = \hat{x}$, θ becomes the angle between $A\hat{x}$ and b

From part b, we have $\cos(A\hat{x}, b) = \dfrac{\|A\hat{x}\|}{\|b\|} = \dfrac{\|A\hat{x}\|}{\|A\hat{x}\|} = 1$

$\Rightarrow \theta$ is 0. (minimum)

Thus the choice $x = \hat{x}$ also minimises ~~the~~ the angle between Ax and b.

6. (10 points) *Gram method for computing least squares approximate solution.* Algorithm 12.1 in the textbook uses the QR factorization to compute the least squares approximate solution $\hat{x} = A^{\dagger}b$, where the $m \times n$ matrix A has linearly independent columns. It has a complexity of $2mn^2$ flops. In this exercise we consider an alternative method: First, form the Gram matrix $G = A^TA$ and the vector $h = A^Tb$; and then compute $\hat{x} = G^{-1}h$ (using algorithm 11.2 in the textbook). What is the complexity of this method? Compare it to algorithm 12.1. *Remark.* You might find that the Gram algorithm appears to be a bit faster than the QR method, but the factor is not large enough to have any practical significance. The idea is useful in situations where G is partially available and can be computed more efficiently than by multiplying A and its transpose.

- Forming the Grom matrix costs approximately mn^2 flops.
- Since we have A is an $m \times n$ matrix and b an m-vector thus h is an n-vector

  For the matrix-vector multiplication A^Tb, since we're doing inner product for each row of A^T with b, which is $2m-1$ flops

  $\Rightarrow$ entire matrix A^T: $n(2m-1) = 2mn$
- Then compute $\hat{x} = G^{-1}h$ using algorithm 11.2 will have a complexity of $2n^3$

$\Rightarrow$ Total flops: $mn^2 + 2mn + 2n^3 \approx mn^2 < 2mn^2$ flops of that of alg. 12.1

since the columns of A are linearly independent $\Rightarrow n \leq m$.

7. (10 points) *Fitting with continuous and discontinuous piecewise-linear functions.* Consider a fitting problem with $n = 1$, so $x^{(1)}, \dots, x^{(N)}$ and $y^{(1)}, \dots, y^{(N)}$ are numbers. We consider two types of closely related models. The first is a piecewise-linear model with knot points at -1 and 1, as described on page 256, and illustrated in figure 13.8. The second is a stratified model (see page 272). with three independent affine models, one for $x < -1$, one for $-1 \le x \le 1$,and one for $x > 1$. (In other words, we stratify on x taking low, middle, or high values.) Are these two models the same? Is one more general than the other? How many parameters does each model have? Hint. See problem title. What can you say about the training set RMS error and test set RMS error that would be achieved using least squares with these two models?

8. (10 points) *Efficient cross-validation.* The cost of fitting a model with p basis functions and N data points (say, using QR factorization) is $2Np^2$ flops. In this exercise we explore the complexity of carrying out 10-fold cross validation on the same data set. We divide the data set into 10 folds, each with $N/10$ data points. The naïve method is to fit 10 different models, each using 9 of the folds, using the QR factorization, which requires $10 \quad 2(0.9)Np^2 = 18Np^2$ flops. (To evaluate each of these models on the remaining fold requires $2(N/10)p$ flops, which can be ignored compared to the cost of fitting the models.) So the naïve method of carrying out 10-fold cross validation requires, not surprisingly, around $10\times$ the number of flops as fitting a single model.

The method below outlines another method to carry out 10-fold cross-validation. Give the total flop count for each step, keeping only the dominant terms, and compare the total cost of the method to that of the naïve method. Let $A_1, \dots, A_{10}$ denote the $(N/10) \times p$ blocks of the data matrix associated with the folds, and let $b_1, \dots, b_{10}$ denote the right-hand sides in the least squares fitting problem.

(a) Form the Gram matrices $G_i = A_i^T A_i$ and the vectors $c_i = A_i^T b_i$.

Each A_i matrix has the shape of $(N/10) \times p$
$\Rightarrow$ Complexity of forming G_i : $\frac{Np^2}{10}$, and ~~&~~ c_i is $\frac{Np}{5}$

(b) Form $G = G_1 + \dots + G_{10}$ and $c = c_1 + \dots + c_{10}$.

Each G_i matrix has the shape of $p \times p$ and c_i is a p-vector
$\Rightarrow$ forming G costs $9p^2$ flops and c_i costs $9p$

(c) For $k = 1, \dots, 10$, compute $\theta_k = (G - G_k)^{-1}(c - c_k)$.

Forming matrix $G - G_k$ costs p^2 flops and $c - c_k$ matrix p flops
Compute θ_k using algorithm 11.2 will cost $2p^3$ flops.