Week 15 Discussion Solutions

CAS CS 132: Geometric Algorithms

December 11, 2023

During discussion sections, we will go over three problems.

- The first will be a warm-up question, to help you verify your understanding of the material.
- The second will be a solution to a problem on the assignment of the previous week.
- The third will be a problem similar to one on the assignment of the following week.

The remainder of the time will be dedicated to open Q&A.

1 LAA 6.6.10 (Warm Up)

Suppose radioactive substances A and B have decay constants of 0.02 and 0.07, respectively. If a mixture of these two substances at time $t = 0$ contains M_A grams of A and M_B grams of B, then a model for the total amount of y of the mixture present at time t is

$$y = M_A e^{-0.2t} + M_B e^{-0.7t}$$

Suppose the initial amounts M_A and M_B are unknown, but a scientist is able to measure the total amounts present at several times and records the following points (t_i, y_i):

$$(10, 21.34), (11, 20.68), (12, 20.05), (14, 18.87), (15, 18.30)$$

Find the design matrix X such that the least squares solution of

$$X \begin{bmatrix} M_A \\ M_B \end{bmatrix} = \begin{bmatrix} 18.30 \\ 18.87 \\ 20.05 \\ 20.68 \\ 21.34 \end{bmatrix}$$

provides an estimate for M_A and M_B.

Solution. The equation

$$M_A e^{-0.2t} + M_B e^{-0.7t}$$

is linear in the parameters M_A and M_B so it is possible to apply the techniques we have seen in class to this problem. The one trick is that we've ordered the observations in increasing order from top to bottom, so we have to make sure we set up our design matrix to reflect this:

$$\begin{bmatrix} e^{-0.2(15)} & e^{-0.7(15)} \\ e^{-0.2(14)} & e^{-0.7(14)} \\ e^{-0.2(12)} & e^{-0.7(12)} \\ e^{-0.2(11)} & e^{-0.7(11)} \\ e^{-0.2(10)} & e^{-0.7(10)} \end{bmatrix} = \begin{bmatrix} e^{-3} & e^{-21/2} \\ e^{-14/5} & e^{-49/5} \\ e^{-12/5} & e^{-42/5} \\ e^{-11/5} & e^{-77/10} \\ e^{-2} & e^{-7} \end{bmatrix}$$

2 Least Square (Homework 11)

Let C be an arbitrary $m \times n$ matrix of rank k and let $\mathbf{b}$ be an arbitrary vector in $\mathbb{R}^m$. Find an expression for the maximum size of a linearly independent set of least squares solutions for $C\mathbf{x} = \mathbf{b}$ where $\mathbf{b} \neq 0$.

Solution. This problem is much about pattern matching and thinking about the process of finding least squares solutions.

A least squares solution satisfies $C\mathbf{x} = \hat{\mathbf{b}}$. So the least squares solutions are solutions to a matrix equation involving C. When we solve the normal equations for $C\mathbf{x} = \mathbf{b}$, we can (per this week's discussion) write the general form solution as

$$\mathbf{v} + x_i\mathbf{w}_i + x_j\mathbf{w}_j + x_k\mathbf{w}_k$$

where x_i, x_j and x_k are free variables (for simplicity, we will assume there are only three). The point here is to recognize is that this is *also* the general form solution to the equation $C\mathbf{x} = \hat{\mathbf{b}}$. In particular, the number of free variables must be the same as **the dimension of the null space of** C, which is $n - k$ by the rank-nullity theorem. Finally, we can recognize that $\{\mathbf{w}_i, \mathbf{w}_j, \mathbf{w}_k\}$ are linearly independent (this is how we generated a basis for the null space) but also $\{\mathbf{v}, \mathbf{w}_i, \mathbf{w}_j, \mathbf{w}_k\}$ is linearly independent since $\mathbf{b} \neq 0$, and so $\hat{\mathbf{b}} \neq 0$ and $C\mathbf{v} = \hat{\mathbf{b}}$ so $\mathbf{v}$ is not $\operatorname{Nul} A$. This set contains $n - k + 1$ vectors, and every least squares solution can be represented as a linear combination of these vectors, so we can have at most $n - k + 1$ linearly independent least squares solutions in a set.

It is worth noting that we'd *like* to talk about dimension here, but solution sets are not subspaces (we demonstrated in a previous assignment that they are affine spaces). But we can still talk about maximal linearly independent sets (recall that the dimension of a subspace U is the maximum size of a linearly independent set of vectors from U).

3 Design Matrices

Write Python code (or pseudocode) which builds the design matrix for modeling with the function

$$f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_k x^k$$

given a feature vector $\mathbf{x}$ (i.e., a vector representing the independent variable for a collection of data points) represented as a 1D NumPy array, and an integer k which determines the degree used in f.

Solution. The approach is to iterate k times the process of adding a new column to the matrix you will eventually return. At a high level:

A. Start with the all-ones vector $A \leftarrow \mathbf{1}$ (`a = np.ones(x.shape[0])`).

B. For i from 1 to k repeat the following: add the vector $\mathbf{x}^k$ to one side of A, i.e., $A \leftarrow [A \;\; \mathbf{x}^k]$. (`a = np.column_stack((a, x ** k))`)

C. return A.