EECS 16A Designing Information Devices and Systems I
Fall 2018 Discussion 14B

1. **(Optional) Orthogonal Matching Pursuit Demo**

   Follow along with your discussion TA.

2. **Pollster: Regularized Least Squares**

   Least squares techniques are useful for many different kinds of prediction problems (also called *regression*). Here, we'll explore how least squares can be used for polling prediction. [1]

   Your job to predict how each county will vote, either for candidate A or candidate B. To do this you will build a linear model that predicts how each county will vote based on the (un)importance of several topics (the economy, healthcare, education, pineapple pizza, etc). Each topic is graded on an importance scale where a negative value means that the topic is not important and a positive value means that the topic is important. The magnitude of the number corresponds to how important/unimportant the topics are to members of the county.

   Each county is represented by a "feature vector" of length 10. The value of each element of the vector captures the importance of that feature to the county. The dataset you are given contains the "feature vectors" for 100 counties as well as how those 100 counties will vote, a value of $+1$ if the county votes for candidate A and a value of -1 if the county votes for candidate B. We want you to find a "good" linear model using the first 90 counties (this is known as training data) and test the "good"ness of your model on the remaining 10 counties (testing data). Your job in this problem is to find the "good" linear model, with weights α_i, using the training data's "feature vectors", $\vec{f}$, and voting decision, b, to minimize the mean square prediction error of the testing data.

   $$b = \alpha_1 f_1 + \alpha_2 f_2 + \ldots + \alpha_{10} f_{10} = \vec{\alpha}^T \vec{f} \tag{1}$$

   (a) **Using only the training data**, set up the least squares problem ($\mathbf{A}\vec{x} = \vec{b}$). How are $\mathbf{A}$, $\vec{x}$, and $\vec{b}$ constructed? What are the dimensions of $\mathbf{A}$, $\vec{x}$ and $\vec{b}$?

   Answer: In this problem the dimensions for training are $N_{examples} = 90$ and $N_{features} = 10$.

   We will create $\mathbf{A}$ by using the equation above written for each training example. We combine them by placing $\vec{f}^T$ for each training example along the columns of $\mathbf{A}$. Because their are 90 training examples and each "feature vector" is length 10. The dimensions of $\mathbf{A}$ will be $N_{examples} \times N_{features} = 90 \times 10$.

   We will generate $\vec{b}$ by placing the voting decision for each county at the index corresponding to its $\vec{f}$ in $\mathbf{A}$. Thus, it will have length $N_{examples} = 90$.

   $\vec{x}$ will contain the linear model weights which we will solve for using linear least squares and is a column vector with length $N_{features} = 10$.

[1] The core ideas we learned in class have been extensively further developed—if you're interested you **should take** EE127A (convex optimization), CS189 (machine learning), EE221 (linear systems)! In these classes you learn ideas that build off of the basic least squares problem for applications in finance, healthcare, advertising, image processing, control, and many other fields.

$$\underbrace{\begin{bmatrix} - & \vec{f}_1 & - \\ - & \vec{f}_2 & - \\ & \vdots & \\ - & \vec{f}_{90} & - \end{bmatrix}}_{\mathbf{A}} \underbrace{\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_9 \\ \alpha_{10} \end{bmatrix}}_{\vec{x}} = \underbrace{\begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{90} \end{bmatrix}}_{\vec{b}} \tag{2}$$

(b) Using IPython, build these matrices and solve for $\hat{\vec{x}}$ using linear least squares with the provided function `doLeastSquares(A,b)`. Evaluate what the mean square prediction error is on the **training** data using your linear model (i.e. $\frac{\|\vec{e}\|^2}{N} = \frac{\|\vec{b}-\mathbf{A}\hat{\vec{x}}\|^2}{N}$). Also, evaluate what the mean square prediction error is on the **testing** data using your linear model.

Answer:

See IPython notebook for exact implementation, but at the high level: Setting up the matrices according to the solution from the previous part and using `doLeastSquares(A,b)` to solve for the feature weights returns roughly

Mean square training prediction error: 0.00828128090803291
Mean square testing prediction error: 0.01026186972414497

(c) A real life problem when building models can be the data itself. In a country with two very polarizing candidates, knowing how a county feels about one topic allows us to predict how important they consider *every* topic. This issue makes the columns of **A** almost linearly dependent—something we observed in the image stitching homework problem and in the imaging lab! Let us analyze it further by looking at the eigenvalues of $\mathbf{A}^T\mathbf{A}$ denoted λ_i. Show that the magnitude of the prediction error,

$$\|\vec{e}\| = \|\vec{b} - \mathbf{A}\hat{\vec{x}}\| = \left\| \vec{b} - \mathbf{A}\left(\frac{\beta_1}{\lambda_1}\vec{v}_1 + \frac{\beta_2}{\lambda_2}\vec{v}_2 + \ldots + \frac{\beta_N}{\lambda_N}\vec{v}_N\right)\right\|. \tag{3}$$

The β_i are the coordinates of the vector $\mathbf{A}^T\vec{b}$ in the eigenbasis of $(\mathbf{A}^T\mathbf{A})^{-1}$,

$$\mathbf{A}^T\vec{b} = \beta_1\vec{v}_1 + \beta_2\vec{v}_2 + \ldots + \beta_N\vec{v}_N \tag{4}$$

(Hint: Consider how the eigenbasis for $(\mathbf{A}^T\mathbf{A})^{-1}$ is related to the eigenbasis of $\mathbf{A}^T\mathbf{A}$). The issue described above will make some of the $\lambda_i \approx 0$. What happens to the magnitude of the prediction error when there exist eigenvalues $\lambda_i \approx 0$? For this problem assume all eigenvalues are distinct and unique.

Answer:

$$\begin{aligned} \|\vec{e}\| &= \|\vec{b} - \mathbf{A}\hat{\vec{x}}\| & (5) \\ &= \|\vec{b} - \mathbf{A}(\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\vec{b}\| & (6) \end{aligned}$$

Now consider expressing $\vec{z} = \mathbf{A}^T\vec{b}$ in the eigenbasis of $(\mathbf{A}^T\mathbf{A})^{-1}$. This will allow us to understand how different components in $\vec{z}$ are affected by applying the matrix $(\mathbf{A}^T\mathbf{A})^{-1}$.

$$
\begin{aligned}
(\mathbf{A}^T\mathbf{A})^{-1}\vec{z} &= (\mathbf{A}^T\mathbf{A})^{-1}(\beta_1\vec{v}_1+\beta_2\vec{v}_2+\ldots) &(7)\\
&= \beta_1(\mathbf{A}^T\mathbf{A})^{-1}\vec{v}_1+\beta_2(\mathbf{A}^T\mathbf{A})^{-1}\vec{v}_2+\ldots &(8)\\
&= \frac{\beta_1}{\lambda_1}\vec{v}_1+\frac{\beta_2}{\lambda_2}\vec{v}_2+\ldots+\frac{\beta_N}{\lambda_N}\vec{v}_N &(9)
\end{aligned}
$$

Plugging back in,

$$
\begin{aligned}
\|\vec{e}\| &= \|\vec{b}-\mathbf{A}\vec{\hat{x}}\| &(10)\\
&= \left\|\vec{b}-\mathbf{A}\left(\frac{\beta_1}{\lambda_1}\vec{v}_1+\frac{\beta_2}{\lambda_2}\vec{v}_2+\ldots+\frac{\beta_N}{\lambda_N}\vec{v}_N\right)\right\| &(11)
\end{aligned}
$$

What this implies is that our mean square prediction error will be large when a $\lambda_i \approx 0$. This is bad and will cause poor prediction performance for individual predictions.

(d) In IPython, plot the eigenvalues of $\mathbf{A}^T\mathbf{A}$. Do we encounter the problem described in part (c)? For this problem, an eigenvalue, λ_i is close to 0 when $\lambda_{max} > 100\lambda_i$.

Answer: See the IPython notebook. After plotting $\mathbf{A}^T\mathbf{A}$'s 10 eigenvalues, the last 6 should be ≈ 0. This will cause the issue described in part (c), i.e. extremely large mean square prediction error.

(e) There are many solutions to this issue, but a common one involves including prior knowledge. We introduce a value γ which we will use to try and rectify the error from part (c). We are given that the weights of our linear model tend to be small (close to zero). Written in equation form,

$$\sqrt{\gamma}\alpha_1 = 0 \ \ldots \ \sqrt{\gamma}\alpha_{10} = 0 \tag{12}$$

and as a matrix,

$$\sqrt{\gamma}\,\mathbf{I}\vec{\alpha} = \vec{0}. \tag{13}$$

Let us concatentate the new equations to the bottom of our matrix equation, $\mathbf{A}\vec{x} = \vec{b}$, from the previous parts to create the augmented matrix, $\tilde{\mathbf{A}}$, and the augmented vector, $\vec{\tilde{b}}$. Our modified matrix equations will be,

$$\tilde{\mathbf{A}}\vec{x} = \vec{\tilde{b}} \rightarrow \left[\begin{array}{c}\mathbf{A}\\ \hline \sqrt{\gamma}\,\mathbf{I}\end{array}\right]\vec{x} = \left[\begin{array}{c}\vec{b}\\ \hline \vec{0}\end{array}\right] \tag{14}$$

What are the dimensions of $\tilde{\mathbf{A}}$, $\vec{\tilde{b}}$? Using IPython, create these matrices using `np.concatenate`.

Answer: We know from part (a) that the dimensions of the original matrix $\mathbf{A}$ was $N_{examples} \times N_{features}$, or 90×10. The size of $\vec{x}$ will not change (because the size of our model, i.e. the amount of α_i values, will not change), so we know that $\sqrt{\gamma}\mathbf{I}$ will be $N_{features} \times N_{features}$. Thus, the dimensions of $\tilde{\mathbf{A}}$ are $(N_{examples}+N_{features}) \times N_{features}$ and the length of $\vec{\tilde{b}}$ is $N_{examples}+N_{features}$.

See IPython notebook for details on implementation with NumPy..

(f) Using the solution to the linear least squares problem derived in class, show that solution to the modified linear least squares problem is

$$\vec{x} = (\tilde{\mathbf{A}}^T\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^T\vec{\tilde{b}} = (\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\mathbf{A}^T\vec{b}. \tag{15}$$

Hint: Matrix-matrix multiplication can be handled in blocks! However, there are restrictions on the dimensions of the matrices when written in this block format, certain dimensions must agree. In this setting, $\mathbf{W} \in \mathbb{R}^{n\times m}$, $\mathbf{Y} \in \mathbb{R}^{m\times l}$, $\mathbf{X} \in \mathbb{R}^{n\times p}$, and $\mathbf{Z} \in \mathbb{R}^{p\times l}$. User beware: Remember that matrix-matrix multiplication does not, in general, commute.

$$\left[\begin{array}{c|c} \mathbf{W} & \mathbf{X} \end{array}\right]\left[\begin{array}{c} \mathbf{Y} \\ \hline \mathbf{Z} \end{array}\right] = \mathbf{WY}+\mathbf{XZ}$$

Answer:

$$\begin{aligned}
\vec{\tilde{x}} &= (\tilde{\mathbf{A}}^T\tilde{\mathbf{A}})^{-1}\tilde{\mathbf{A}}^T\vec{\tilde{b}} &(16)\\
&= \left(\left[\begin{array}{c}\mathbf{A}\\ \hline \sqrt{\gamma}\,\mathbf{I}\end{array}\right]^T\left[\begin{array}{c}\mathbf{A}\\ \hline \sqrt{\gamma}\,\mathbf{I}\end{array}\right]\right)^{-1}\left[\begin{array}{c}\mathbf{A}\\ \hline \sqrt{\gamma}\,\mathbf{I}\end{array}\right]^T\left[\begin{array}{c}\vec{b}\\ \hline \vec{0}\end{array}\right] &(17)\\
&= \left(\left[\begin{array}{c|c}\mathbf{A}^T & \sqrt{\gamma}\,\mathbf{I}\end{array}\right]\left[\begin{array}{c}\mathbf{A}\\ \hline \sqrt{\gamma}\,\mathbf{I}\end{array}\right]\right)^{-1}\left[\begin{array}{c|c}\mathbf{A}^T & \sqrt{\gamma}\,\mathbf{I}\end{array}\right]\left[\begin{array}{c}\vec{b}\\ \hline \vec{0}\end{array}\right] &(18)\\
&= (\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}(\mathbf{A}^T\vec{b}+\sqrt{\gamma}\,\mathbf{I}\vec{0}) &(19)\\
&= (\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\mathbf{A}^T\vec{b} &(20)
\end{aligned}$$

(g) Show that the new total prediction error

$$\|\vec{e}\| = \|\vec{b}-\mathbf{A}\vec{\tilde{x}}\| = \left\|\vec{b}-\mathbf{A}\left(\frac{\beta_1}{\lambda_1+\gamma}\vec{v}_1+\frac{\beta_2}{\lambda_2+\gamma}\vec{v}_2+\ldots+\frac{\beta_N}{\lambda_N+\gamma}\vec{v}_N\right)\right\|, \tag{21}$$

using our modified least squares solution, $\vec{\tilde{x}} = (\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\mathbf{A}^T\vec{b}$, the eigenvalues λ_i of $\mathbf{A}^T\mathbf{A}$, and the eigenvectors $\vec{v}_i$ of $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$. The β_i are the coordinates of the vector $\mathbf{A}^T\vec{b}$ in the eigenbasis of $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$,

$$\mathbf{A}^T\vec{b} = \beta_1\vec{v}_1+\beta_2\vec{v}_2+\ldots+\beta_N\vec{v}_N \tag{22}$$

(Hint: Consider how the eigenbasis for $\mathbf{A}^T\mathbf{A}+\gamma I$ is related to the eigenbasis of $\mathbf{A}^T\mathbf{A}$. Think about diagonalization of $\mathbf{A}^T\mathbf{A}$.) For this problem assume all eigenvalues are distinct and unique.

Answer:

$$\begin{aligned}
\|\vec{e}\| &= \|\vec{b}-\mathbf{A}\vec{\tilde{x}}\| &(23)\\
&= \|\vec{b}-\mathbf{A}(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\mathbf{A}^T\vec{b}\| &(24)
\end{aligned}$$

Let us calculate the eigenvalues of $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$ in terms of the eigenvalues of $\mathbf{A}^T\mathbf{A}$. First, we will calculate the eigenvalues of $\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I}$ in terms of the eigenvalues of $\mathbf{A}^T\mathbf{A}$ by considering its diagonalization, $\mathbf{U}\Lambda\mathbf{U}^{-1}$,

$$\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I} = \mathbf{U}\Lambda\mathbf{U}^{-1}+\gamma\mathbf{U}\mathbf{U}^{-1} = \mathbf{U}(\Lambda+\gamma\mathbf{I})\mathbf{U}^{-1} \tag{25}$$

.
The eigenvalues of $\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I}$ are $\lambda_i+\gamma$ and the eigenvalues of $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$ are simply $\frac{1}{\lambda_i+\gamma}$.

Going through a similar process as the previous part, we express $\vec{z}=\mathbf{A}^T\vec{b}$ in the eigenbasis of $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$. This will allow us to understand how different components in $\vec{z}$ are affected by applying the matrix $(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}$.

$$\begin{aligned}(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\vec{z} &= (\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}(\beta_1\vec{v}_1+\beta_2\vec{v}_2+\dots) &(26)\\ &= \beta_1(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\vec{v}_1+\beta_2(\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I})^{-1}\vec{v}_2+\dots &(27)\\ &= \frac{\beta_1}{\lambda_1+\gamma}\vec{v}_1+\frac{\beta_2}{\lambda_2+\gamma}\vec{v}_2+\dots &(28)\end{aligned}$$

Plugging back in,

$$\begin{aligned}\|\vec{e}\| &= \|\vec{b}-\mathbf{A}\vec{\hat{x}}\| &(29)\\ &= \left\|\vec{b}-\mathbf{A}\left(\frac{\beta_1}{\lambda_1+\gamma}\vec{v}_1+\frac{\beta_2}{\lambda_2+\gamma}\vec{v}_2+\dots+\frac{\beta_N}{\lambda_N+\gamma}\vec{v}_N\right)\right\| &(30)\end{aligned}$$

Wow that is incredible!!!(Say this out loud) By adding a small modification to the least squares problem, the prediction error now depends on the $\lambda_i+\gamma$ rather than λ_i on its own. We don't have control over λ_i, but we *do* have control over γ, meaning when we have $\lambda_i\approx 0$, we can control the error by setting $\gamma>0$!

As one might imagine, there is no free lunch. We cannot set γ to any value, and the next parts of the problem explores how to find the best γ.

(h) In IPython, in **a single plot** display the eigenvalues of $\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I}$ for several values of γ (e.g. 0, 10, 100). What does this do to the eigenvalues of $\mathbf{A}^T\mathbf{A}$?

Answer: The eigenvalues of $\mathbf{A}^T\mathbf{A}+\gamma\mathbf{I}$ are shifted up by γ, i.e. the new eigenvalues $\mu_i=\lambda_i+\gamma$

(i) In IPython, let us now find the "best" γ to improve our testing total prediction error. Evalute the total testing prediction error using different values of γ (use the list in the IPython Notebook). What is the best choice of γ (ie. which gamma minimizes the total testing prediction error)? How does the total testing prediction error using modified least squares with the best choice of γ compare with the total testing prediction error from part (b)?

Answer: See IPython notebook.

Optimal gamma: 9.326033468832199
Achieved Mean Square Prediction Error: 0.009394364837293731

The curve should dip down and then back up. There is a γ that minimizes the total testing prediction error. The total testing prediction error in the part will be less than the total testing prediction error from part (b).

If this problem whets your appetite, awesome! You should take classes in the areas of signal processing, optimization, and machine learning: EE127A, EE120, EE123, CS189, EE221, and EE225A.

3. One Magical Procedure (Fall 2015 Final)

This problem will be gone over during linear algebra review session.

Suppose that we have a vector $\vec{x} \in \mathbb{R}^5$ and an $N \times 5$ measurement matrix $\mathbf{M}$ defined by column vectors $\vec{c}_1, \ldots, \vec{c}_5$, such that:

$$\mathbf{M}\vec{x} = \begin{bmatrix} | & & | \\ \vec{c}_1 & \cdots & \vec{c}_5 \\ | & & | \end{bmatrix} \vec{x} \approx \vec{b}$$

We can treat the vector $\vec{b} \in \mathbb{R}^N$ as a noisy measurement of the vector $\vec{x}$, with measurement matrix $\mathbf{M}$ and some additional noise in it as well.

You also know that the true $\vec{x}$ is sparse – it only has two non-zero entries and all the rest of the entries are zero in reality. Our goal is to recover this original $\vec{x}$ as best we can.

However, your intern has managed to lose not only the measurements $\vec{b}$ but the entire measurement matrix $\mathbf{M}$ as well!

Fortunately, you have found a backup in which you have all the pairwise inner products $\langle \vec{c}_i, \vec{c}_j \rangle$ between the columns of $\mathbf{M}$ and each other as well as all the inner products $\left\langle \vec{c}_i, \vec{b} \right\rangle$ between the columns of $\mathbf{M}$ and the vector $\vec{b}$. Finally, you also know the inner product $\left\langle \vec{b}, \vec{b} \right\rangle$ of $\vec{b}$ with itself.

All the information you have is captured in the following table of inner products. (These are not the vectors themselves.)

$\langle \cdot, \cdot \rangle$	$\vec{c}_1$	$\vec{c}_2$	$\vec{c}_3$	$\vec{c}_4$	$\vec{c}_5$	$\vec{b}$
$\vec{c}_1$	2	0	1	-1	1	1
$\vec{c}_2$		2	1	-1	-1	-5
$\vec{c}_3$			2	0	-1	2
$\vec{c}_4$				2	-1	6
$\vec{c}_5$					2	-1
$\vec{b}$						29

(So, for example, if you read this table, you will see that the inner product $\langle \vec{c}_2, \vec{c}_3 \rangle = 1$, that the inner product $\left\langle \vec{c}_3, \vec{b} \right\rangle = 2$, and that the inner product $\left\langle \vec{b}, \vec{b} \right\rangle = 29$. By symmetry of the real inner product, $\langle \vec{c}_3, \vec{c}_2 \rangle = 1$ as well.)

Your goal is to find which entries of $\vec{x}$ are non-zero and what their values are.

(a) Use the information in the table above to answer which of the $\vec{c}_1, \ldots, \vec{c}_5$ has the largest magnitude inner product with $\vec{b}$.

Answer:

Reading off the table, $\boxed{\vec{c}_4}$ has the largest inner product with $\vec{b}$.

(b) Let the vector with the largest magnitude inner product with $\vec{b}$ be $\vec{c}_a$. Let $\vec{b}_p$ be the projection of $\vec{b}$ onto $\vec{c}_a$. Write $\vec{b}_p$ symbolically as an expression only involving $\vec{c}_a$, $\vec{b}$, and their inner products with themselves and each other.

Answer:

The magnitude of the projection is $\frac{\langle \vec{c}_a, \vec{b} \rangle}{\|\vec{c}_a\|}$, and the direction of the projection is $\frac{\vec{c}_a}{\|\vec{c}_a\|}$. Thus:

$$\vec{b}_p = \boxed{\frac{\left\langle \vec{c}_a, \vec{b} \right\rangle}{\langle \vec{c}_a, \vec{c}_a \rangle} \vec{c}_a}$$

(c) Use the information in the table above to find which of the column vectors $\vec{c}_1, \ldots, \vec{c}_5$ has the largest magnitude inner product with the residue $\vec{b} - \vec{b}_p$.
Hint: The linearity of inner products might prove useful.
Answer:
The inner product of $\vec{b} - \vec{b}_p$ with a vector $\vec{c}_i$ is:

$$\left\langle \vec{b} - \vec{b}_p, \vec{c}_i \right\rangle = \left\langle \vec{b}, \vec{c}_i \right\rangle - \frac{\left\langle \vec{c}_a, \vec{b} \right\rangle}{\langle \vec{c}_a, \vec{c}_a \rangle} \langle \vec{c}_a, \vec{c}_i \rangle$$

Finding the numerical values of the inner products:

$\left\langle \vec{b} - \vec{b}_p, \vec{c}_1 \right\rangle$	$\left\langle \vec{b} - \vec{b}_p, \vec{c}_2 \right\rangle$	$\left\langle \vec{b} - \vec{b}_p, \vec{c}_3 \right\rangle$	$\left\langle \vec{b} - \vec{b}_p, \vec{c}_4 \right\rangle$	$\left\langle \vec{b} - \vec{b}_p, \vec{c}_5 \right\rangle$
4	-2	2	0	2

Thus the vector with the highest inner product with the residue is: $\boxed{\vec{c}_1}$.

(d) Suppose that the vectors we found in parts (a) and (c) are $\vec{c}_a$ and $\vec{c}_c$. These correspond to the components of $\vec{x}$ that are non-zero, that is, $\vec{b} \approx x_a \vec{c}_a + x_c \vec{c}_c$. However, there might be noise in the measurements $\vec{b}$, so we want to find the linear least squares estimates $\widehat{x}_a$ and $\widehat{x}_c$. Write a matrix expression for $\begin{bmatrix} \widehat{x}_a \\ \widehat{x}_c \end{bmatrix}$ in terms of appropriate matrices filled with the inner products of $\vec{c}_a$, $\vec{c}_c$, $\vec{b}$.
Answer:
We use least squares to solve for $\begin{bmatrix} \widehat{x}_a \\ \widehat{x}_c \end{bmatrix}$. Let $\mathbf{A} = \begin{bmatrix} \vec{c}_a & \vec{c}_c \end{bmatrix}$. Using the least-squares formula,

$$\begin{aligned} \begin{bmatrix} \widehat{x}_a \\ \widehat{x}_c \end{bmatrix} &= (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \vec{b} \\ &= \begin{bmatrix} \langle \vec{c}_a, \vec{c}_a \rangle & \langle \vec{c}_a, \vec{c}_c \rangle \\ \langle \vec{c}_c, \vec{c}_a \rangle & \langle \vec{c}_c, \vec{c}_c \rangle \end{bmatrix}^{-1} \begin{bmatrix} \left\langle \vec{c}_a, \vec{b} \right\rangle \\ \left\langle \vec{c}_c, \vec{b} \right\rangle \end{bmatrix} \end{aligned}$$

(e) Compute the numerical values of $\widehat{x}_a$ and $\widehat{x}_c$ using the information in the table.
Answer:
Substituting the previous expression with values from the table, we get: $x_1 = 2\frac{2}{3}, x_4 = 4\frac{1}{3}$.

$$\begin{aligned} \begin{bmatrix} \widehat{x}_4 \\ \widehat{x}_1 \end{bmatrix} &= \begin{bmatrix} \langle \vec{c}_4, \vec{c}_4 \rangle & \langle \vec{c}_4, \vec{c}_1 \rangle \\ \langle \vec{c}_1, \vec{c}_4 \rangle & \langle \vec{c}_1, \vec{c}_1 \rangle \end{bmatrix}^{-1} \begin{bmatrix} \left\langle \vec{c}_4, \vec{b} \right\rangle \\ \left\langle \vec{c}_1, \vec{b} \right\rangle \end{bmatrix} \\ &= \begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 6 \\ 1 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \begin{bmatrix} 6 \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{13}{3} \\ \frac{8}{3} \end{bmatrix} \end{aligned}$$