

Orthogonal Projection, Low Rank Approximation, and Orthogonal Bases

11.1 Opening Remarks

11.1.1 Low Rank Approximation



11.1.2 Outline

11.1. Opening Remarks	463
11.1.1. Low Rank Approximation	463
11.1.2. Outline	464
11.1.3. What You Will Learn	465
11.2. Projecting a Vector onto a Subspace	466
11.2.1. Component in the Direction of ...	466
11.2.2. An Application: Rank-1 Approximation	470
11.2.3. Projection onto a Subspace	474
11.2.4. An Application: Rank-2 Approximation	476
11.2.5. An Application: Rank-k Approximation	478
11.3. Orthonormal Bases	481
11.3.1. The Unit Basis Vectors, Again	481
11.3.2. Orthonormal Vectors	482
11.3.3. Orthogonal Bases	485
11.3.4. Orthogonal Bases (Alternative Explanation)	488
11.3.5. The QR Factorization	492
11.3.6. Solving the Linear Least-Squares Problem via QR Factorization	493
11.3.7. The QR Factorization (Again)	494
11.4. Change of Basis	498
11.4.1. The Unit Basis Vectors, One More Time	498
11.4.2. Change of Basis	498
11.5. Singular Value Decomposition	501
11.5.1. The Best Low Rank Approximation	501
11.6. Enrichment	505
11.6.1. The Problem with Computing the QR Factorization	505
11.6.2. QR Factorization Via Householder Transformations (Reflections)	505
11.6.3. More on SVD	505
11.7. Wrap Up	506
11.7.1. Homework	506
11.7.2. Summary	506

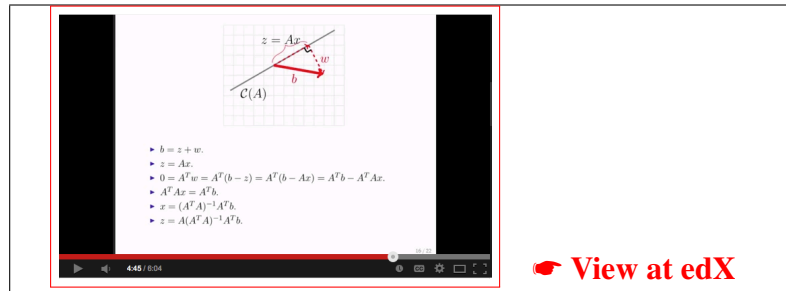
11.1.3 What You Will Learn

Upon completion of this unit, you should be able to

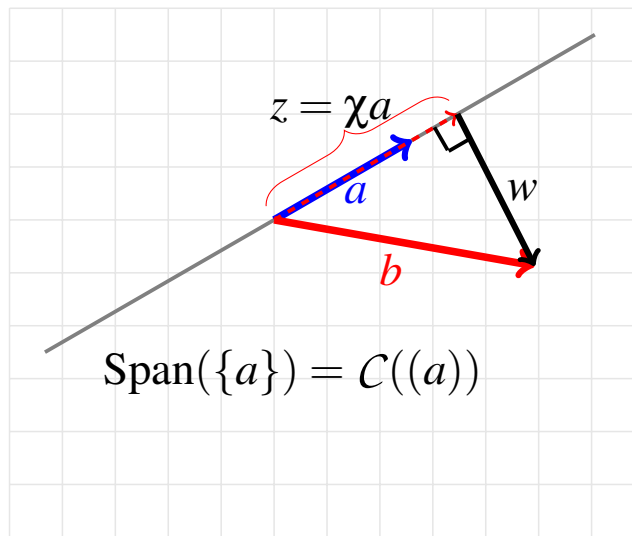
- Given vectors a and b in \mathbb{R}^m , find the component of b in the direction of a and the component of b orthogonal to a .
 - Given a matrix A with linear independent columns, find the matrix that projects any given vector b onto the column space A and the matrix that projects b onto the space orthogonal to the column space of A , which is also called the left null space of A .
 - Understand low rank approximation, projecting onto columns to create a rank- k approximation.
 - Identify, apply, and prove simple properties of orthonormal vectors.
 - Determine if a set of vectors is orthonormal.
 - Transform a set of basis vectors into an orthonormal basis using Gram-Schmidt orthogonalization.
 - Compute an orthonormal basis for the column space of A .
 - Apply Gram-Schmidt orthogonalization to compute the QR factorization.
 - Solve the Linear Least-Squares Problem via the QR Factorization.
 - Make a change of basis.
 - Be aware of the existence of the Singular Value Decomposition and that it provides the “best” rank- k approximation.
-

11.2 Projecting a Vector onto a Subspace

11.2.1 Component in the Direction of ...



Consider the following picture:



Here, we have two vectors, $a, b \in \mathbb{R}^m$. They exist in the plane defined by $\text{Span}(\{a, b\})$ which is a two dimensional space (unless a and b point in the same direction). From the picture, we can also see that b can be thought of as having a component z in the direction of a and another component w that is orthogonal (perpendicular) to a . The component in the direction of a lies in the $\text{Span}(\{a\}) = C((a))$ (here (a) denotes the matrix with only once column, a) while the component that is orthogonal to a lies in $\text{Span}(\{a\})^\perp$. Thus,

$$b = z + w,$$

where

- $z = \chi a$ with $\chi \in \mathbb{R}$; and
- $a^T w = 0$.

Noting that $w = b - z$ we find that

$$0 = a^T w = a^T (b - z) = a^T (b - \chi a)$$

or, equivalently,

$$a^T a \chi = a^T b.$$

We have seen this before. Recall that when you want to approximately solve $Ax = b$ where b is not in $C(A)$ via Linear Least Squares, the “best” solution satisfies $A^T Ax = A^T b$. The equation that we just derived is the exact same, except that A has one column: $A = (a)$.

Then, provided $a \neq 0$,

$$\chi = (a^T a)^{-1} (a^T b).$$

Thus, the component of b in the direction of a is given by

$$u = \chi a = (a^T a)^{-1} (a^T b) a = a (a^T a)^{-1} (a^T b) = [a (a^T a)^{-1} a^T] b.$$

Note that we were able to move a to the left of the equation because $(a^T a)^{-1}$ and $a^T b$ are both scalars. The component of b orthogonal (perpendicular) to a is given by

$$w = b - z = b - (a (a^T a)^{-1} a^T) b = I b - (a (a^T a)^{-1} a^T) b = (I - a (a^T a)^{-1} a^T) b.$$

Summarizing:

$$\begin{aligned} z &= (a (a^T a)^{-1} a^T) b && \text{is the component of } b \text{ in the direction of } a; \text{ and} \\ w &= (I - a (a^T a)^{-1} a^T) b && \text{is the component of } b \text{ perpendicular (orthogonal) to } a. \end{aligned}$$

We say that, given vector a , the matrix that *projects* any given vector b onto the space spanned by a is given by

$$a (a^T a)^{-1} a^T \quad (= \frac{1}{a^T a} a a^T)$$

since $a (a^T a)^{-1} a^T b$ is the component of b in $\text{Span}(\{a\})$. Notice that this is an outer product:

$$a \underbrace{(a^T a)^{-1} a^T}_{v^T}.$$

We say that, given vector a , the matrix that *projects* any given vector b onto the space orthogonal to the space spanned by a is given by

$$I - a (a^T a)^{-1} a^T \quad (= I - \frac{1}{a^T a} a a^T = I - a v^T),$$

since $(I - a (a^T a)^{-1} a^T) b$ is the component of b in $\text{Span}(\{a\})^\perp$.

Notice that $I - \frac{1}{a^T a} a a^T = I - a v^T$ is a rank-1 update to the identity matrix.

Homework 11.2.1.1 Let $a = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^\perp$, respectively. Compute

1. $P_a\left(\begin{pmatrix} 2 \\ 0 \end{pmatrix}\right) =$

2. $P_a^\perp\left(\begin{pmatrix} 2 \\ 0 \end{pmatrix}\right) =$

3. $P_a\left(\begin{pmatrix} 4 \\ 2 \end{pmatrix}\right) =$

4. $P_a^\perp\left(\begin{pmatrix} 4 \\ 2 \end{pmatrix}\right) =$

5. Draw a picture for each of the above.

Homework 11.2.1.2 Let $a = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}$ and $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto

$\text{Span}(\{a\})$ and $\text{Span}(\{a\})^\perp$, respectively. Compute

$$1. P_a\left(\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}\right) =$$

$$2. P_a^\perp\left(\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}\right) =$$

$$3. P_a\left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right) =$$

$$4. P_a^\perp\left(\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}\right) =$$

Homework 11.2.1.3 Let $a, v, b \in \mathbb{R}^m$.

What is the approximate cost of computing $(av^T)b$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m^2$.
- $2m^2 + 4m$.

What is the approximate cost of computing $(v^Tb)a$, obeying the order indicated by the parentheses?

- $m^2 + 2m$.
- $3m$.
- $2m^2 + 4m$.

For computational efficiency, it is important to compute $a(a^T a)^{-1} a^T b$ according to order indicated by the following parentheses:

$$((a^T a)^{-1} (a^T b)) a.$$

Similarly, $(I - a(a^T a)^{-1} a^T) b$ should be computed as

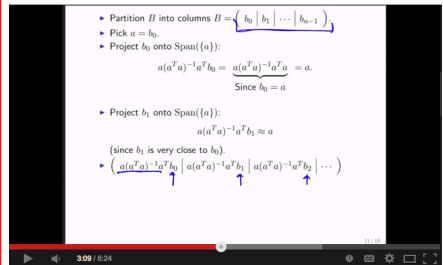
$$b - (((a^T a)^{-1} (a^T b)) a).$$

Homework 11.2.1.4 Given $a, x \in \mathbb{R}^m$, let $P_a(x)$ and $P_a^\perp(x)$ be the projection of vector x onto $\text{Span}(\{a\})$ and $\text{Span}(\{a\})^\perp$, respectively. Then which of the following are true:

- | | |
|---|------------|
| 1. $P_a(a) = a$. | True/False |
| 2. $P_a(\chi a) = \chi a$. | True/False |
| 3. $P_a^\perp(\chi a) = 0$ (the zero vector). | True/False |
| 4. $P_a(P_a(x)) = P_a(x)$. | True/False |
| 5. $P_a^\perp(P_a^\perp(x)) = P_a^\perp(x)$. | True/False |
| 6. $P_a(P_a^\perp(x)) = 0$ (the zero vector). | True/False |

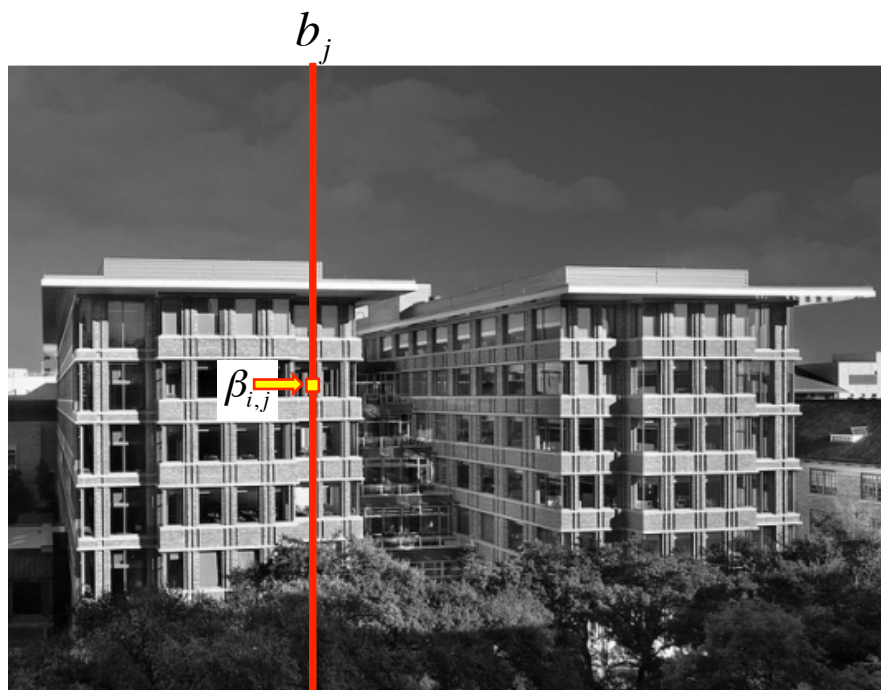
(Hint: Draw yourself a picture.)

11.2.2 An Application: Rank-1 Approximation



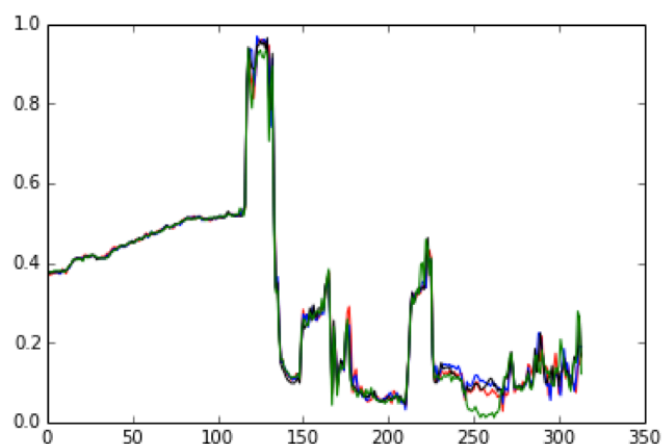
[View at edX](#)

Consider the picture



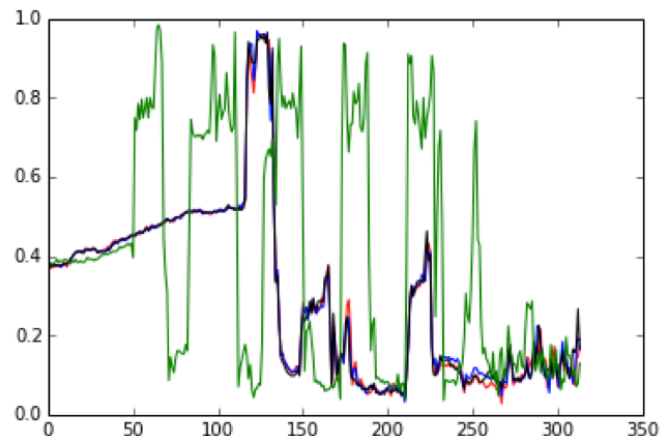
This picture can be thought of as a matrix $B \in \mathbb{R}^{m \times n}$ where each element in the matrix encodes a pixel in the picture. The j th column of B then encodes the j th column of pixels in the picture.

Now, let's focus on the first few columns. Notice that there is a lot of similarity in those columns. This can be illustrated by plotting the values in the column as a function of the element in the column:

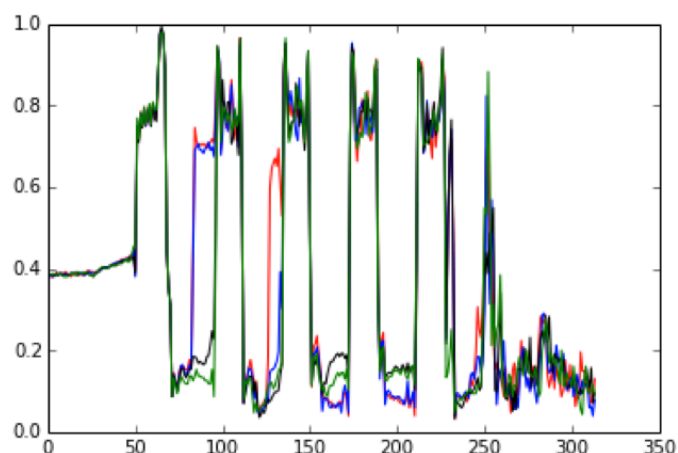


In the graph on the left, we plot $\beta_{i,j}$, the value of the (i, j) pixel, for $j = 0, 1, 2, 3$ in different colors. The picture on the right highlights the columns for which we are doing this. The green line corresponds to $j = 3$ and you notice that it is starting to deviate some for i near 250.

If we now instead look at columns $j = 0, 1, 2, 100$, where the green line corresponds to $j = 100$, we see that that column in the picture is dramatically different:



Changing this to plotting $j = 100, 101, 102, 103$ and we notice a lot of similarity again:



Now, let's think about this from the point of view taking one vector, say the first column of B , b_0 , and projecting the other columns onto the span of that column. What does this mean?

- Partition B into columns $B = \left(b_0 \mid b_1 \mid \cdots \mid b_{n-1} \right)$.
- Pick $a = b_0$.
- Focus on projecting b_0 onto $\text{Span}(\{a\})$:

$$a(a^T a)^{-1} a^T b_0 = \underbrace{a(a^T a)^{-1} a^T a}_{\text{Since } b_0 = a} = a.$$

Of course, this is what we expect when projecting a vector onto itself.

- Next, focus on projecting b_1 onto $\text{Span}(\{a\})$:

$$a(a^T a)^{-1} a^T b_1$$

since b_1 is very close to b_0 .

- Do this for all columns, and create a picture with all of the projected vectors:

$$\left(a(a^T a)^{-1} a^T b_0 \mid a(a^T a)^{-1} a^T b_1 \mid a(a^T a)^{-1} a^T b_2 \mid \cdots \right)$$

- Now, remember that if T is some matrix, then

$$TB = \left(Tb_0 \mid Tb_1 \mid Tb_2 \mid \cdots \right).$$

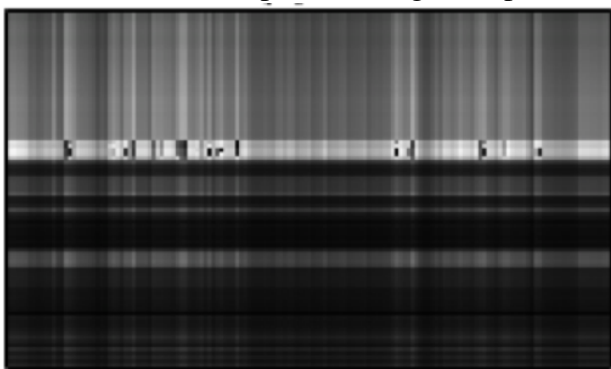
If we let $T = a(a^T a)^{-1} a^T$ (the matrix that projects onto $\text{Span}(\{a\})$), then

$$a(a^T a)^{-1} a^T \left(b_0 \mid b_1 \mid b_2 \mid \cdots \right) = a(a^T a)^{-1} a^T B.$$

- We can manipulate this further by recognizing that $y^T = (a^T a)^{-1} a^T B$ can be computed as $y = (a^T a)^{-1} B^T a$:

$$a(a^T a)^{-1} a^T B = a \underbrace{((a^T a)^{-1} B^T a)}_y = ay^T$$

- We now recognize ay^T as an outer product (a column vector times a row vector).
- If we do this for our picture, we get the picture on the left:



Notice how it seems like each column is the same, except with some constant change in the gray-scale. The same is true for rows. Why is this? If you focus on the left-most columns in the picture, they almost look correct (comparing to the left-most columns in the picture on the right). Why is this?

- The benefit of the approximation on the left is that it can be described with two vectors: a and y ($n + m$ floating point numbers) while the original matrix on the right required an entire matrix ($m \times n$ floating point numbers).
- The disadvantage of the approximation on the left is that it is hard to recognize the original picture...

Homework 11.2.2.1 Let \mathbf{S} and \mathbf{T} be subspaces of \mathbb{R}^m and $\mathbf{S} \subset \mathbf{T}$.
 $\dim(\mathbf{S}) \leq \dim(\mathbf{T})$.

Always/Sometimes/Never

Homework 11.2.2.2 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then the $m \times n$ matrix uv^T has a rank of at most one.

True/False

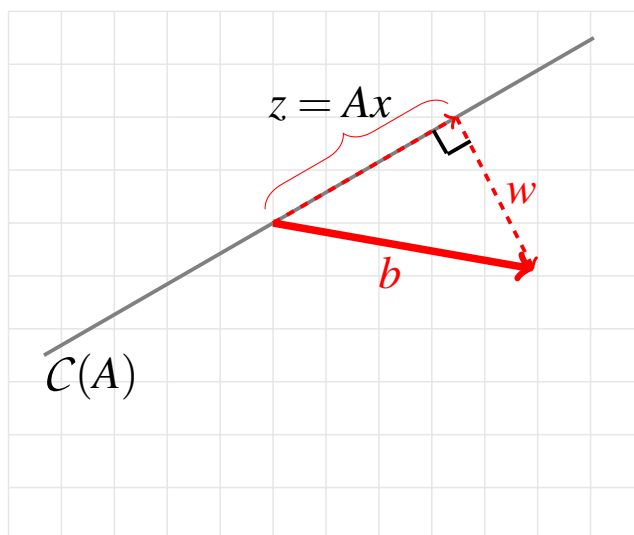
Homework 11.2.2.3 Let $u \in \mathbb{R}^m$ and $v \in \mathbb{R}^n$. Then uv^T has rank equal to zero if (Mark all correct answers.)

1. $u = 0$ (the zero vector in \mathbb{R}^m).
2. $v = 0$ (the zero vector in \mathbb{R}^n).
3. Never.
4. Always.

11.2.3 Projection onto a Subspace

No video this section

Next, consider the following picture:



What we have here are

- Matrix $A \in \mathbb{R}^{m \times n}$.
- The space spanned by the columns of A : $C(A)$.
- A vector $b \in \mathbb{R}^m$.
- Vector z , the component of b in $C(A)$ which is also the vector in $C(A)$ closest to the vector b . Since this vector is in the column space of A , $z = Ax$ for some vector $x \in \mathbb{R}^n$.
- The vector w which is the component of b orthogonal to $C(A)$.

The vectors b, z, w , all exist in the same planar subspace since $b = z + w$, which is the page on which these vectors are drawn in the above picture.

Thus,

$$b = z + w,$$

where

- $z = Ax$ with $x \in \mathbb{R}^n$; and
- $A^T w = 0$ since w is orthogonal to the column space of A and hence in $\mathcal{N}(A^T)$.

Noting that $w = b - z$ we find that

$$0 = A^T w = A^T (b - z) = A^T (b - Ax)$$

or, equivalently,

$$A^T Ax = A^T b.$$

This should look familiar!

Then, provided $(A^T A)^{-1}$ exists (which, we saw before happens when A has linearly independent columns),

$$x = (A^T A)^{-1} A^T b.$$

Thus, the component of b in $\mathcal{C}(A)$ is given by

$$z = Ax = A(A^T A)^{-1} A^T b$$

while the component of b orthogonal (perpendicular) to $\mathcal{C}(A)$ is given by

$$w = b - z = b - A(A^T A)^{-1} A^T b = Ib - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T) b.$$

Summarizing:

$$\begin{aligned} z &= A(A^T A)^{-1} A^T b \\ w &= (I - A(A^T A)^{-1} A^T) b. \end{aligned}$$

We say that, given matrix A with linearly independent columns, the matrix that *projects* a given vector b onto the column space of A is given by

$$A(A^T A)^{-1} A^T$$

since $A(A^T A)^{-1} A^T b$ is the component of b in $\mathcal{C}(A)$.

We say that, given matrix A with linearly independent columns, the matrix that *projects* a given vector b onto the space orthogonal to the column space of A (which, recall, is the *left null space* of A) is given by

$$I - A(A^T A)^{-1} A^T$$

since $(I - A(A^T A)^{-1} A^T) b$ is the component of b in $\mathcal{C}(A)^\perp = \mathcal{N}(A^T)$.

Homework 11.2.3.1 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$ and $b = \begin{pmatrix} 1 \\ 2 \\ 7 \end{pmatrix}$.

1. Find the projection of b onto the column space of A .
2. Split b into $z + w$ where z is in the column space and w is perpendicular (orthogonal) to that space.
3. Which of the four subspaces ($C(A)$, $R(A)$, $\mathcal{N}(A)$, $\mathcal{N}(A^T)$) contains w ?

For computational reasons, it is important to compute $A(A^T A)^{-1} A^T x$ according to order indicated by the following parentheses:

$$A[(A^T A)^{-1}[A^T x]]$$

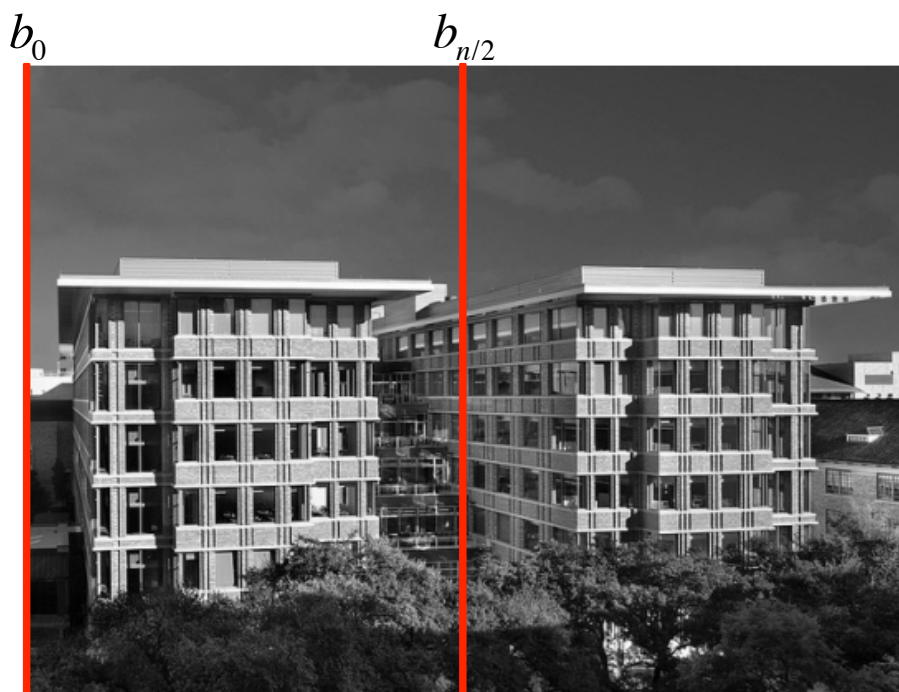
Similarly, $(I - A(A^T A)^{-1} A^T)x$ should be computed as

$$x - [A[(A^T A)^{-1}[A^T x]]]$$

11.2.4 An Application: Rank-2 Approximation



Earlier, we took the first column as being representative of all columns of the picture. Looking at the picture, this is clearly not the case. But what if we took two columns instead, say column $j = 0$ and $j = n/2$, and projected each of the columns onto the subspace spanned by those two columns:



- Partition B into columns $B = \left(b_0 \mid b_1 \mid \cdots \mid b_{n-1} \right)$.

- Pick $A = \left(a_0 \mid a_1 \right) = \left(b_0 \mid b_{n/2} \right)$.

- Focus on projecting b_0 onto $\text{Span}(\{a_0, a_1\}) = C(A)$:

$$A(A^T A)^{-1} A^T b_0 = a = b_0$$

because a is in $C(A)$ and a is therefore the best vector in $C(A)$.

- Next, focus on projecting b_1 onto $\text{Span}(\{a\})$:

$$A(A^T A)^{-1} A^T b_1 \approx b_1$$

since b_1 is very close to a .

- Do this for all columns, and create a picture with all of the projected vectors:

$$\left(A(A^T A)^{-1} A^T b_0 \mid A(A^T A)^{-1} A^T b_1 \mid A(A^T A)^{-1} A^T b_2 \mid \cdots \right)$$

- Now, remember that if T is some matrix, then

$$TB = \left(Tb_0 \mid Tb_1 \mid Tb_2 \mid \cdots \right).$$

If we let $T = A(A^T A)^{-1} A^T$ (the matrix that projects onto $C(A)$), then

$$A(A^T A)^{-1} A^T \left(b_0 \mid b_1 \mid b_2 \mid \cdots \right) = A(A^T A)^{-1} A^T B.$$

- We can manipulate this by letting $W = B^T A (A^T A)^{-1}$ so that

$$A \underbrace{(A^T A)^{-1} A^T B}_{W^T} = A W^T.$$

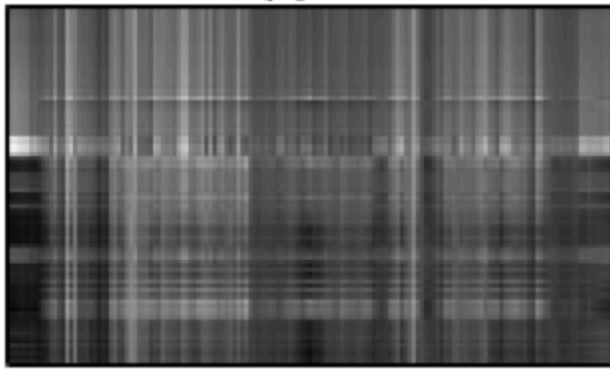
Notice that A and W each have two columns.

- We now recognize $A W^T$ is the sum of two outer products:

$$A W^T = \begin{pmatrix} a_0 & a_1 \end{pmatrix} \begin{pmatrix} w_0 & w_1 \end{pmatrix}^T = \begin{pmatrix} a_0 & a_1 \end{pmatrix} \begin{pmatrix} w_0^T \\ w_1^T \end{pmatrix} = a_0 w_0^T + a_1 w_1^T.$$

It can be easily shown that this matrix has rank of at most two, which is why this would be called a rank-2 approximation of B .

- If we do this for our picture, we get the picture on the left:



We are starting to see some more detail.

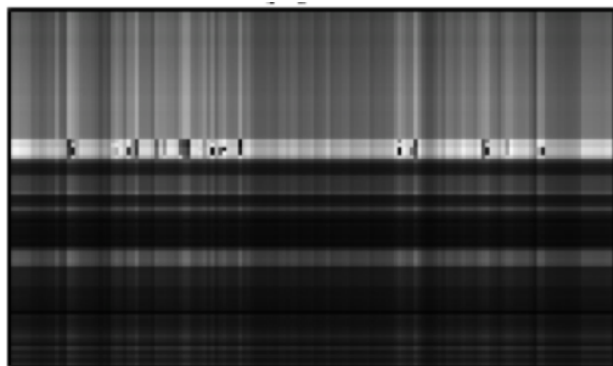
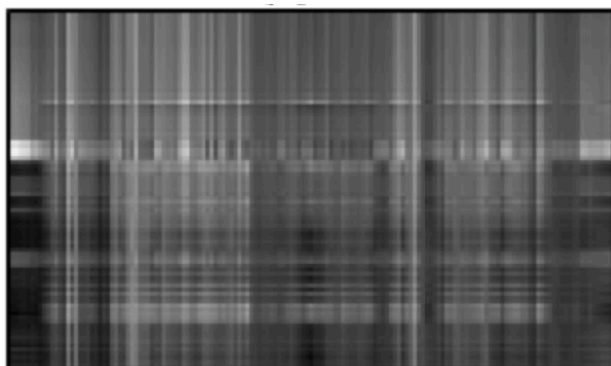
- We now have to store only a $n \times 2$ and $m \times 2$ matrix (A and W).

11.2.5 An Application: Rank-k Approximation



Rank-k approximations

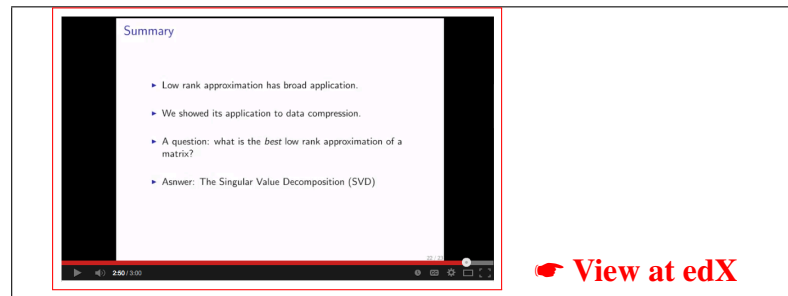
We can improve the approximations above by picking progressively more columns for A . The following progression of pictures shows the improvement as more and more columns are used, where k indicates the number of columns:

 $k = 1$  $k = 2$  $k = 10$  $k = 25$  $k = 50$ 

original

Homework 11.2.5.1 Let $U \in \mathbb{R}^{m \times k}$ and $V \in \mathbb{R}^{n \times k}$. Then the $m \times n$ matrix UV^T has rank at most k .

True/False



Homework 11.2.5.2 We discussed in this section that the projection of B onto the column space of A is given by $A(A^T A)^{-1} A^T B$. So, if we compute $V = (A^T A)^{-1} A^T B$, then AV is an approximation to B that requires only $m \times k$ matrix A and $k \times n$ matrix V .

To compute V , we can perform the following steps:

- Form $C = A^T A$.
- Compute the LU factorization of C , overwriting C with the resulting L and U .
- Compute $V = A^T B$.
- Solve $LX = V$, overwriting V with the solution matrix X .
- Solve $UX = V$, overwriting V with the solution matrix X .
- Compute the approximation of B as $A \cdot V$ (A times V). In practice, you would not compute this approximation, but store A and V instead, which typically means less data is stored.

To experiment with this, download **Week11.zip**, place it in

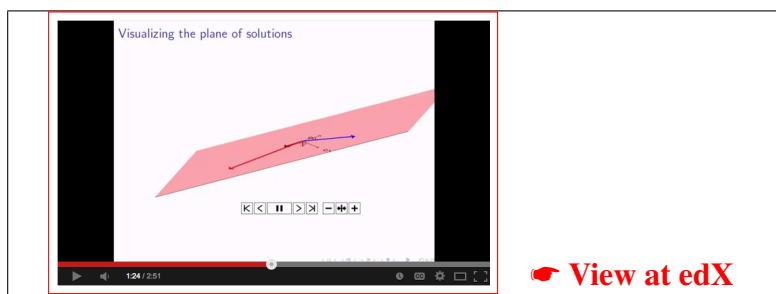
LAFFSpring2015 -> Programming

and unzip it. Then examine the file `Week11/CompressPicture.m`, look for the comments on what operations need to be inserted, and insert them. Execute the script in the Command Window and see how the picture in file `building.png` is approximated. Play with the number of columns used to approximate. Find your own picture! (It will have to be a black-and-white picture for what we discussed to work.)

Notice that $A^T A$ is a symmetric matrix, and it can be shown to be symmetric positive definite under most circumstances (when A has linearly independent columns). This means that instead of the LU factorization, one can use the Cholesky factorization (see the enrichment in Week 8). In `Week11.zip` you will also find a function for computing the Cholesky factorization. Try to use it to perform the calculations.

11.3 Orthonormal Bases

11.3.1 The Unit Basis Vectors, Again



Recall the unit basis vectors in \mathbb{R}^3 :

$$e_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad e_2 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}.$$

This set of vectors forms a basis for \mathbb{R}^3 ; they are linearly independent and any vector $x \in \mathbb{R}^3$ can be written as a linear combination of these three vectors.

Now, the set

$$v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

is also a basis for \mathbb{R}^3 , but is not nearly as nice:

- Two of the vectors are not of length one.
- They are not orthogonal to each other.

There is something pleasing about a basis that is **orthonormal**. By this we mean that each vector in the basis is of length one, and any pair of vectors is orthogonal to each other.

A question we are going to answer in the next few units is how to take a given basis for a subspace and create an orthonormal basis from it.

Homework 11.3.1.1 Consider the vectors

$$v_0 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad v_1 = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \quad \text{and} \quad v_2 = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

1. Compute

(a) $v_0^T v_1 =$

(b) $v_0^T v_2 =$

(c) $v_1^T v_2 =$

2. These vectors are orthonormal. True/False

11.3.2 Orthonormal Vectors



Definition 11.1 Let $q_0, q_1, \dots, q_{k-1} \in \mathbb{R}^m$. Then these vectors are (mutually) orthonormal if for all $0 \leq i, j < k$:

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

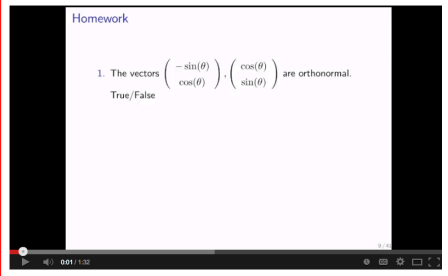
Homework 11.3.2.1

1. $\begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & -\sin(\theta) \\ \sin(\theta) & \cos(\theta) \end{pmatrix} =$

2. $\begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix}^T \begin{pmatrix} \cos(\theta) & \sin(\theta) \\ -\sin(\theta) & \cos(\theta) \end{pmatrix} =$

3. The vectors $\begin{pmatrix} -\sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ \sin(\theta) \end{pmatrix}$ are orthonormal. True/False

4. The vectors $\begin{pmatrix} \sin(\theta) \\ \cos(\theta) \end{pmatrix}, \begin{pmatrix} \cos(\theta) \\ -\sin(\theta) \end{pmatrix}$ are orthonormal. True/False



[View at edX](#)

Homework 11.3.2.2 Let $q_0, q_1, \dots, q_{k-1} \in \mathbb{R}^m$ be a set of orthonormal vectors. Let

$$Q = \left(q_0 \mid q_1 \mid \cdots \mid q_{k-1} \right).$$

Then $Q^T Q = I$.

TRUE/FALSE



[View at edX](#)

Homework 11.3.2.3 Let $Q \in \mathbb{R}^{m \times k}$ (with $k \leq m$) and $Q^T Q = I$. Partition

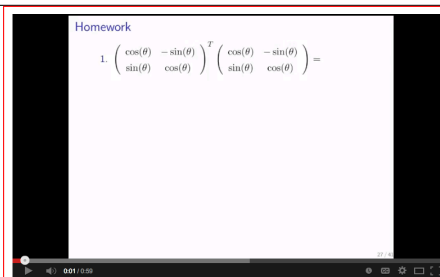
$$Q = \left(q_0 \mid q_1 \mid \cdots \mid q_{k-1} \right).$$

Then q_0, q_1, \dots, q_{k-1} are orthonormal vectors.

TRUE/FALSE



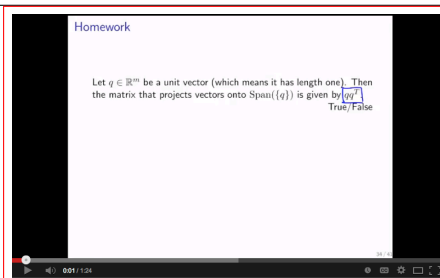
[View at edX](#)



[View at edX](#)

Homework 11.3.2.4 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Then the matrix that projects vectors onto $\text{Span}(\{q\})$ is given by qq^T .

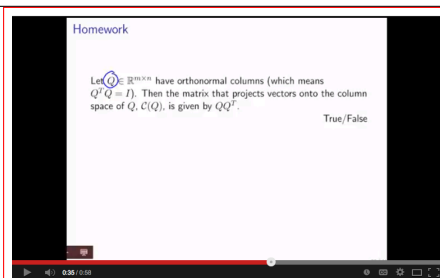
True/False



[View at edX](#)

Homework 11.3.2.5 Let $q \in \mathbb{R}^m$ be a unit vector (which means it has length one). Let $x \in \mathbb{R}^m$. Then the component of x in the direction of q (in $\text{Span}(\{q\})$) is given by $q^T x q$.

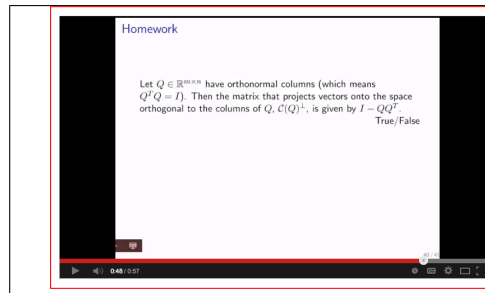
True/False



[View at edX](#)

Homework 11.3.2.6 Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the column space of Q , $C(Q)$, is given by QQ^T .

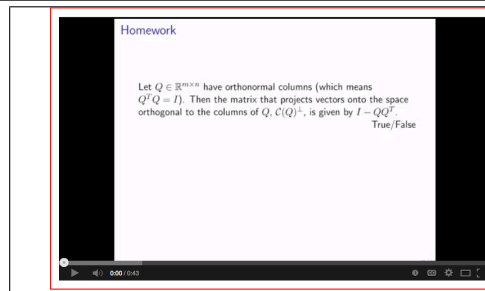
True/False



[View at edX](#)

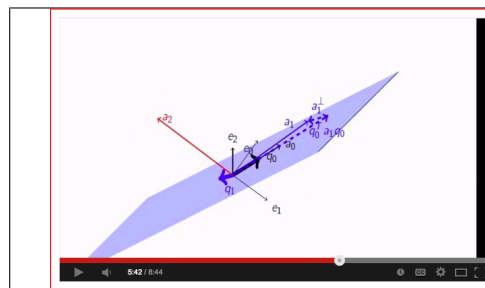
Homework 11.3.2.7 Let $Q \in \mathbb{R}^{m \times n}$ have orthonormal columns (which means $Q^T Q = I$). Then the matrix that projects vectors onto the space orthogonal to the columns of Q , $C(Q)^\perp$, is given by $I - QQ^T$.

True/False

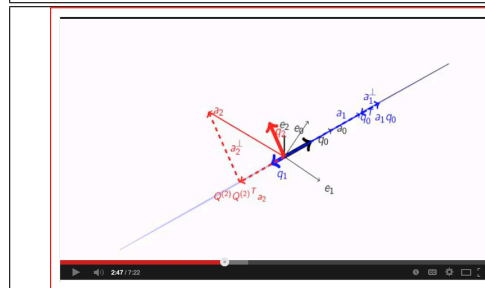


[View at edX](#)

11.3.3 Orthogonal Bases



[View at edX](#)



[View at edX](#)

The fundamental idea for this unit is that it is convenient for a basis to be orthonormal. The question is: how do we transform a given set of basis vectors (e.g., the columns of a matrix A with linearly independent columns) into a set of orthonormal vectors that form a basis for the same space? The process we will described is known as **Gram-Schmidt orthogonalization** (GS orthogonalization).

The idea is very simple:

- Start with a set of n linearly independent vectors, $a_0, a_1, \dots, a_{n-1} \in \mathbb{R}^m$.
- Take the first vector and make it of unit length:

$$q_0 = a_0 / \underbrace{\|a_0\|_2}_{\rho_{0,0}},$$

where $\rho_{0,0} = \|a_0\|_2$, the length of a_0 .

Notice that $\text{Span}(\{a_0\}) = \text{Span}(\{q_0\})$ since q_0 is simply a scalar multiple of a_0 .

This gives us one orthonormal vector, q_0 .

- Take the second vector, a_1 , and compute its component *orthogonal* to q_0 :

$$a_1^\perp = (I - q_0 q_0^T) a_1 = a_1 - q_0 q_0^T a_1 = a_1 - \underbrace{q_0^T a_1}_{\rho_{0,1}} q_0.$$

- Take a_1^\perp , the component of a_1 *orthogonal* to q_0 , and make it of unit length:

$$q_1 = a_1^\perp / \underbrace{\|a_1^\perp\|_2}_{\rho_{1,1}},$$

We will see later that $\text{Span}(\{a_0, a_1\}) = \text{Span}(\{q_0, q_1\})$.

This gives us two orthonormal vectors, q_0, q_1 .

- Take the third vector, a_2 , and compute its component *orthogonal* to $Q^{(2)} = \begin{pmatrix} q_0 & q_1 \end{pmatrix}$ (orthogonal to both q_0 and q_1 and hence $\text{Span}(\{q_0, q_1\}) = \mathcal{C}(Q^{(2)})$):

$$\begin{aligned} a_2^\perp &= \underbrace{(I - Q^{(2)} Q^{(2)T}) a_2}_{\substack{\text{Projection} \\ \text{onto } \mathcal{C}(Q^{(2)})^\perp}} = a_2 - \underbrace{Q^{(2)} Q^{(2)T} a_2}_{\substack{\text{Component} \\ \text{in } \mathcal{C}(Q^{(2)})}} = a_2 - \begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} q_0 & q_1 \end{pmatrix}^T a_2 \\ &= a_2 - \begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} q_0^T \\ q_1^T \end{pmatrix} a_2 = a_2 - \begin{pmatrix} q_0 & q_1 \end{pmatrix} \begin{pmatrix} q_0^T a_2 \\ q_1^T a_2 \end{pmatrix} \\ &= a_2 - (q_0^T a_2 q_0 + q_1^T a_2 q_1) \\ &= a_2 - \underbrace{q_0^T a_2 q_0}_{\substack{\text{Component} \\ \text{in direction} \\ \text{of } q_0}} - \underbrace{q_1^T a_2 q_1}_{\substack{\text{Component} \\ \text{in direction} \\ \text{of } q_1}}. \end{aligned}$$

Notice:

- $a_2 - q_0^T a_2 q_0$ equals the vector a_2 with the component in the direction of q_0 subtracted out.
- $a_2 - q_0^T a_2 q_0 - q_1^T a_2 q_1$ equals the vector a_2 with the components in the direction of q_0 **and** q_1 subtracted out.
- Thus, a_2^\perp equals component of a_2 that is orthogonal to both q_0 **and** q_1 .
- Take a_2^\perp , the component of a_2 *orthogonal* to q_0 and q_1 , and make it of unit length:

$$q_2 = a_2^\perp / \underbrace{\|a_2^\perp\|_2}_{\rho_{2,2}},$$

We will see later that $\text{Span}(\{a_0, a_1, a_2\}) = \text{Span}(\{q_0, q_1, q_2\})$.

This gives us three orthonormal vectors, q_0, q_1, q_2 .

- (Continue repeating the process)
- Take vector a_k , and compute its component *orthogonal* to $Q^{(k)} = \begin{pmatrix} q_0 & q_1 & \cdots & q_{k-1} \end{pmatrix}$ (orthogonal to all vectors q_0, q_1, \dots, q_{k-1} and hence $\text{Span}(\{q_0, q_1, \dots, q_{k-1}\}) = C(Q^{(k)})$):

$$\begin{aligned} a_k^\perp &= (I - Q^{(k)} Q^{(k)T}) a_k = a_k - Q^{(k)} Q^{(k)T} a_k = a_k - \begin{pmatrix} q_0 & q_1 & \cdots & q_{k-1} \end{pmatrix} \begin{pmatrix} q_0 & q_1 & \cdots & q_{k-1} \end{pmatrix}^T a_k \\ &= a_k - \begin{pmatrix} q_0 & q_1 & \cdots & q_{k-1} \end{pmatrix} \begin{pmatrix} q_0^T \\ q_1^T \\ \vdots \\ q_{k-1}^T \end{pmatrix} a_k = a_k - \begin{pmatrix} q_0 & q_1 & \cdots & q_{k-1} \end{pmatrix} \begin{pmatrix} q_0^T a_k \\ q_1^T a_k \\ \vdots \\ q_{k-1}^T a_k \end{pmatrix} \\ &= a_k - q_0^T a_k q_0 - q_1^T a_k q_1 - \cdots - q_{k-1}^T a_k q_{k-1}. \end{aligned}$$

Notice:

- $a_k - q_0^T a_k q_0$ equals the vector a_k with the component in the direction of q_0 subtracted out.
- $a_k - q_0^T a_k q_0 - q_1^T a_k q_1$ equals the vector a_k with the components in the direction of q_0 **and** q_1 subtracted out.
- $a_k - q_0^T a_k q_0 - q_1^T a_k q_1 - \cdots - q_{k-1}^T a_k q_{k-1}$ equals the vector a_k with the components in the direction of q_0, q_1, \dots, q_{k-1} subtracted out.
- Thus, a_k^\perp equals component of a_k that is orthogonal to all vectors q_j that have already been computed.
- Take a_k^\perp , the component of a_k *orthogonal* to q_0, q_1, \dots, q_{k-1} , and make it of unit length:

$$q_k = a_k^\perp / \underbrace{\|a_k^\perp\|_2}_{\rho_{k,k}},$$

We will see later that $\text{Span}(\{a_0, a_1, \dots, a_k\}) = \text{Span}(\{q_0, q_1, \dots, q_k\})$.

This gives us $k + 1$ orthonormal vectors, q_0, q_1, \dots, q_k .

- Continue this process to compute q_0, q_1, \dots, q_{n-1} .

The following result is the whole point of the Gram-Schmidt process, namely to find an orthonormal basis for the span of a given set of linearly independent vectors.

Theorem 11.2 *Let $a_0, a_1, \dots, a_{k-1} \in \mathbb{R}^m$ be linearly independent vectors and let $q_0, q_1, \dots, q_{k-1} \in \mathbb{R}^m$ be the result of Gram-Schmidt orthogonalization. Then $\text{Span}(\{a_0, a_1, \dots, a_{k-1}\}) = \text{Span}(\{q_0, q_1, \dots, q_{k-1}\})$.*

The proof is a bit tricky (and in some sense stated in the material in this unit) so we do not give it here.

11.3.4 Orthogonal Bases (Alternative Explanation)

Computing q_1

- $\text{Span}(\{a_0, a_1\}) = \text{Span}(\{q_0, q_1\})$
- $a_1 = \rho_{0,1}q_0 + \rho_{1,1}q_1$
- $q_0^T q_1 = 0$ and $q_1^T q_1 = 1$
- $q_0^T a_1 = q_0^T (\rho_{0,1}q_0 + \rho_{1,1}q_1) = \rho_{0,1}q_0^T q_0 + \rho_{1,1}q_0^T q_1 = \rho_{0,1} \cdot 1 + \rho_{1,1} \cdot 0 = \rho_{0,1}$
- $\rho_{1,1}q_1 = a_1 - \rho_{0,1}q_0$
- $q_1 = \frac{a_1 - \rho_{0,1}q_0}{\rho_{1,1}}$

$\rho_{0,1} := q_0^T a_1$
 $a_1^+ := a_1 - \rho_{0,1}q_0$
 $\rho_{1,1} := \|a_1^+\|_2$
 $q_1 := a_1^+ / \rho_{1,1}$

[View at edX](#)

We now give an alternate explanation for Gram-Schmidt orthogonalization.

We are given linearly independent vectors $a_0, a_1, \dots, a_{n-1} \in \mathbb{R}^m$ and would like to compute orthonormal vectors $q_0, q_1, \dots, q_{n-1} \in \mathbb{R}^m$ such that $\text{Span}(\{a_0, a_1, \dots, a_{n-1}\})$ equals $\text{Span}(\{q_0, q_1, \dots, q_{n-1}\})$.

Let's put one more condition on the vectors q_k : $\text{Span}(\{a_0, a_1, \dots, a_{k-1}\}) = \text{Span}(\{q_0, q_1, \dots, q_{k-1}\})$ for $k = 0, 1, \dots, n$. In other words,

$$\begin{aligned}
 \text{Span}(\{a_0\}) &= \text{Span}(\{q_0\}) \\
 \text{Span}(\{a_0, a_1\}) &= \text{Span}(\{q_0, q_1\}) \\
 &\vdots \\
 \text{Span}(\{a_0, a_1, \dots, a_{k-1}\}) &= \text{Span}(\{q_0, q_1, \dots, q_{k-1}\}) \\
 &\vdots \\
 \text{Span}(\{a_0, a_1, \dots, a_{n-1}\}) &= \text{Span}(\{q_0, q_1, \dots, q_{n-1}\})
 \end{aligned}$$

Computing q_0

Now, $\text{Span}(\{a_0\}) = \text{Span}(\{q_0\})$ means that $a_0 = \rho_{0,0}q_0$ for some scalar $\rho_{0,0}$. Since q_0 has to be of length one, we can choose

$$\begin{aligned}
 \rho_{0,0} &:= \|a_0\|_2 \\
 q_0 &:= a_0 / \rho_{0,0}.
 \end{aligned}$$

Notice that q_0 is not unique: we could have chosen $\rho_{0,0} = -\|a_0\|_2$ and $q_0 = a_0 / \rho_{0,0}$. This non-uniqueness is recurring in the below discussion, and we will ignore it since we are merely interested in a *single* orthonormal basis.

Computing q_1

Next, we note that $\text{Span}(\{a_0, a_1\}) = \text{Span}(\{q_0, q_1\})$ means that $a_1 = \rho_{0,1}q_0 + \rho_{1,1}q_1$ for some scalars $\rho_{0,1}$ and $\rho_{1,1}$. We also know that $q_0^T q_1 = 0$ and $q_1^T q_1 = 1$ since these vectors are orthonormal. Now

$$q_0^T a_1 = q_0^T (\rho_{0,1}q_0 + \rho_{1,1}q_1) = q_0^T \rho_{0,1}q_0 + q_0^T \rho_{1,1}q_1 = \rho_{0,1} \underbrace{q_0^T q_0}_1 + \rho_{1,1} \underbrace{q_0^T q_1}_0 = \rho_{0,1}$$

so that

$$\rho_{0,1} = q_0^T a_1.$$

Once $\rho_{0,1}$ has been computed, we can compute the component of a_1 orthogonal to q_0 :

$$\underbrace{\rho_{1,1}q_1}_{a_1^\perp} = a_1 - \underbrace{q_0^T a_1}_{\rho_{0,1}} q_0$$

after which $a_1^\perp = \rho_{1,1}q_1$. Again, we can now compute $\rho_{1,1}$ as the length of a_1^\perp and normalize to compute q_1 :

$$\begin{aligned} \rho_{0,1} &:= q_0^T a_1 \\ a_1^\perp &:= a_1 - \rho_{0,1}q_0 \\ \rho_{1,1} &:= \|a_1^\perp\|_2 \\ q_1 &:= a_1^\perp / \rho_{1,1}. \end{aligned}$$

Computing q_2

We note that $\text{Span}(\{a_0, a_1, a_2\}) = \text{Span}(\{q_0, q_1, q_2\})$ means that $a_2 = \rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2$ for some scalars $\rho_{0,2}$, $\rho_{1,2}$ and $\rho_{2,2}$. We also know that $q_0^T q_2 = 0$, $q_1^T q_2 = 0$ and $q_2^T q_2 = 1$ since these vectors are orthonormal. Now

•

$$q_0^T a_2 = q_0^T (\rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2) = \rho_{0,2} \underbrace{q_0^T q_0}_1 + \rho_{1,2} \underbrace{q_0^T q_1}_0 + \rho_{2,2} \underbrace{q_0^T q_2}_0 = \rho_{0,2}$$

so that

$$\rho_{0,2} = q_0^T a_2.$$

•

$$q_1^T a_2 = q_1^T (\rho_{0,2}q_0 + \rho_{1,2}q_1 + \rho_{2,2}q_2) = \rho_{0,2} \underbrace{q_1^T q_0}_0 + \rho_{1,2} \underbrace{q_1^T q_1}_1 + \rho_{2,2} \underbrace{q_1^T q_2}_0 = \rho_{1,2}$$

so that

$$\rho_{1,2} = q_1^T a_2.$$

Once $\rho_{0,2}$ and $\rho_{1,2}$ have been computed, we can compute the component of a_2 orthogonal to q_0 and q_1 :

$$\underbrace{\rho_{2,2}q_2}_{a_2^\perp} = a_2 - \underbrace{q_0^T a_2}_{\rho_{0,2}} q_0 - \underbrace{q_1^T a_2}_{\rho_{1,2}} q_1$$

after which $a_2^\perp = \rho_{2,2}q_2$. Again, we can now compute $\rho_{2,2}$ as the length of a_2^\perp and normalize to compute q_2 :

$$\begin{aligned}\rho_{0,2} &:= q_0^T a_2 \\ \rho_{1,2} &:= q_1^T a_2 \\ a_2^\perp &:= a_2 - \rho_{0,2}q_0 - \rho_{1,2}q_1 \\ \rho_{2,2} &:= \|a_2^\perp\|_2 \\ q_2 &:= a_2^\perp / \rho_{2,2}.\end{aligned}$$

Computing q_k

Let's generalize this: $\text{Span}(\{a_0, a_1, \dots, a_k\}) = \text{Span}(\{q_0, q_1, \dots, q_k\})$ means that

$$a_k = \rho_{0,k}q_0 + \rho_{1,k}q_1 + \dots + \rho_{k-1,k}q_{k-1} + \rho_{k,k}q_k = \sum_{j=0}^{k-1} \rho_{j,k}q_j + \rho_{k,k}q_k$$

for some scalars $\rho_{0,k}, \rho_{1,k}, \dots, \rho_{k,k}$. We also know that

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Now, if $p < k$,

$$q_p^T a_k = q_p^T \left(\sum_{j=0}^{k-1} \rho_{j,k}q_j + \rho_{k,k}q_k \right) = \sum_{j=0}^{k-1} \rho_{j,k}q_p^T q_j + \rho_{k,k}q_p^T q_k = \rho_{p,k}q_p^T q_p = \rho_{p,k}$$

so that

$$\rho_{p,k} = q_p^T a_k.$$

Once the scalars $\rho_{p,k}$ have been computed, we can compute the component of a_k orthogonal to q_0, \dots, q_{k-1} :

$$\underbrace{\rho_{k,k}q_k}_{a_k^\perp} = a_k - \sum_{j=0}^{k-1} \underbrace{q_j^T a_k}_{\rho_{j,k}} q_j$$

after which $a_k^\perp = \rho_{k,k}q_k$. Once again, we can now compute $\rho_{k,k}$ as the length of a_k^\perp and normalize to compute q_k :

$$\rho_{0,k} := q_0^T a_k$$

$$\begin{aligned}
& \vdots \\
\rho_{k-1,k} &:= q_{k-1}^T a_k \\
a_k^\perp &:= a_k - \sum_{j=0}^{k-1} \rho_{j,k} q_j \\
\rho_{k,k} &:= \|a_k^\perp\|_2 \\
q_k &:= a_k^\perp / \rho_{k,k}.
\end{aligned}$$

An algorithm

The above discussion yields an algorithm for Gram-Schmidt orthogonalization, computing q_0, \dots, q_{n-1} (and all the $\rho_{i,j}$'s as a side product). This is not a FLAME algorithm so it may take longer to comprehend:

for $k = 0, \dots, n-1$

$$\left. \begin{array}{l} \textbf{for } p = 0, \dots, k-1 \\ \rho_{p,k} := q_p^T a_k \\ \textbf{endfor} \end{array} \right\} \left(\begin{array}{c} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{array} \right) = \left(\begin{array}{c} q_0^T a_k \\ q_1^T a_k \\ \vdots \\ q_{k-1}^T a_k \end{array} \right) = \left(\begin{array}{c} q_0^T \\ q_1^T \\ \vdots \\ q_{k-1}^T \end{array} \right) a_k = \left(q_0 \mid q_1 \mid \dots \mid q_{k-1} \right)^T a_k$$

$$\left. \begin{array}{l} a_k^\perp := a_k \\ \textbf{for } j = 0, \dots, k-1 \\ a_k^\perp := a_k^\perp - \rho_{j,k} q_j \\ \textbf{endfor} \end{array} \right\} a_k^\perp = a_k - \sum_{j=0}^{k-1} \rho_{j,k} q_j = a_k - \left(q_0 \mid q_1 \mid \dots \mid q_{k-1} \right) \left(\begin{array}{c} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{array} \right)$$

$$\left. \begin{array}{l} \rho_{k,k} := \|a_k^\perp\|_2 \\ q_k := a_k^\perp / \rho_{k,k} \end{array} \right\} \text{Normalize } a_k^\perp \text{ to be of length one.}$$

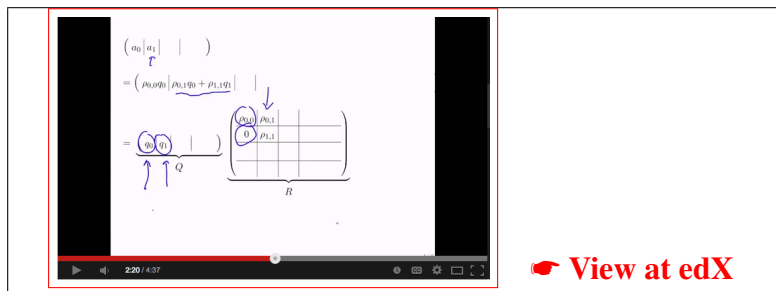
endfor

Homework 11.3.4.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ Compute an orthonormal basis for $\mathcal{C}(A)$.

Homework 11.3.4.2 Consider $A = \begin{pmatrix} 1 & -1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 2 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

Homework 11.3.4.3 Consider $A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute an orthonormal basis for $\mathcal{C}(A)$.

11.3.5 The QR Factorization



Given linearly independent vectors $a_0, a_1, \dots, a_{n-1} \in \mathbb{R}^m$, the last unit computed the orthonormal basis q_0, q_1, \dots, q_{n-1} such that $\text{Span}(\{a_1, a_2, \dots, a_{n-1}\})$ equals $\text{Span}(\{q_1, q_2, \dots, q_{n-1}\})$. As a side product, the scalars $\rho_{i,j} = q_i^T a_j$ were computed, for $i \leq j$. We now show that in the process we computed what's known as the **QR factorization** of the matrix $A = \begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}$:

$$\underbrace{\begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}}_A = \underbrace{\begin{pmatrix} q_0 & q_1 & \cdots & q_{n-1} \end{pmatrix}}_Q \underbrace{\begin{pmatrix} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_{n-1,n-1} \end{pmatrix}}_R.$$

Notice that $Q^T Q = I$ (since its columns are orthonormal) and R is upper triangular.

In the last unit, we noticed that

$$\begin{aligned} a_0 &= \rho_{0,0} q_0 \\ a_1 &= \rho_{0,1} q_0 + \rho_{1,1} q_1 \\ \vdots &\quad \vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ a_{n-1} &= \rho_{0,n-1} q_0 + \rho_{1,n-1} q_1 + \cdots + \rho_{n-1,n-1} q_{n-1} \end{aligned}$$

If we write the vectors on the left of the equal signs as the columns of a matrix, and do the same for the vectors on the right of the equal signs, we get

$$\underbrace{\begin{pmatrix} a_0 & a_1 & \cdots & a_{n-1} \end{pmatrix}}_A = \begin{pmatrix} \rho_{0,0} q_0 & \rho_{0,1} q_0 + \rho_{1,1} q_1 & \cdots & \rho_{0,n-1} q_0 + \rho_{1,n-1} q_1 + \cdots + \rho_{n-1,n-1} q_{n-1} \end{pmatrix}$$

$$= \underbrace{\begin{pmatrix} q_0 & q_1 & \cdots & q_{n-1} \end{pmatrix}}_Q \underbrace{\begin{pmatrix} \rho_{0,0} & \rho_{0,1} & \cdots & \rho_{0,n-1} \\ 0 & \rho_{1,1} & \cdots & \rho_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \rho_{n-1,n-1} \end{pmatrix}}_R.$$

Bingo, we have shown how Gram-Schmidt orthogonalization computes the QR factorization of a matrix A .

Homework 11.3.5.1 Consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$.

- Compute the QR factorization of this matrix.
(Hint: Look at Homework 11.3.4.1)
- Check that $QR = A$.

Homework 11.3.5.2 Consider x !m

$A = \begin{pmatrix} 1 & 1 \\ 1 & -1 \\ -2 & 4 \end{pmatrix}$. Compute the QR factorization of this matrix.

(Hint: Look at Homework 11.3.4.3)

Check that $A = QR$.

11.3.6 Solving the Linear Least-Squares Problem via QR Factorization

[View at edX](#)

Now, let's look at how to use the QR factorization to solve $Ax \approx b$ when b is not in the column space of A but A has linearly independent columns. We know that the linear least-squares solution is given by

$$x = (A^T A)^{-1} A^T b.$$

Now $A = QR$ where $Q^T Q = I$. Then

$$x = (A^T A)^{-1} A^T b = \left(\underbrace{(QR)^T}_A \underbrace{(QR)}_A \right)^{-1} \underbrace{(QR)^T}_A b$$

$$\begin{aligned}
 &= (R^T \underbrace{Q^T Q}_I R)^{-1} R^T Q^T b = (R^T R)^{-1} R^T Q^T b = R^{-1} \underbrace{R^{-T} R^T}_I Q^T b \\
 &= R^{-1} Q^T b.
 \end{aligned}$$

Thus, the linear least-square solution, x , for $Ax \approx b$ when A has linearly independent columns solves $Rx = Q^T b$.

Homework 11.3.6.1 In Homework 11.3.4.1 you were asked to consider $A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix}$ and

compute an orthonormal basis for $\mathcal{C}(A)$.

In Homework 11.3.5.1 you were then asked to compute the QR factorization of that matrix. Of course, you could/should have used the results from Homework 11.3.4.1 to save yourself calculations. The result was the following factorization $A = QR$:

$$\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array} \right) = \left(\begin{array}{c|c} \frac{1}{\sqrt{2}} & \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \\ \frac{\sqrt{2}}{\sqrt{3}} & \begin{pmatrix} -\frac{1}{2} \\ 1 \\ \frac{1}{2} \end{pmatrix} \end{array} \right) \left(\begin{array}{c|c} \sqrt{2} & \frac{1}{\sqrt{2}} \\ 0 & \frac{\sqrt{6}}{2} \end{array} \right)$$

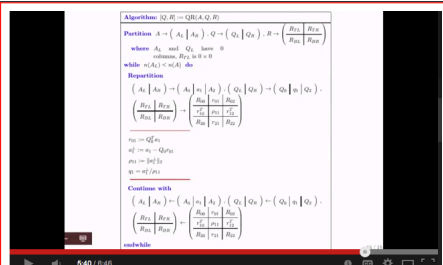
Now, compute the “best” solution (in the linear least-squares sense), \hat{x} , to

$$\left(\begin{array}{c|c} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{array} \right) \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix}.$$

(This is the same problem as in Homework 10.4.2.1.)

- $u = Q^T b =$
- The solution to $R\hat{x} = u$ is $\hat{x} =$

11.3.7 The QR Factorization (Again)



[View at edX](#)

We now give an explanation of how to compute the QR factorization that yields an algorithm in FLAME notation.

We wish to compute $A = QR$ where $A, Q \in \mathbb{R}^{m \times n}$ and $R \in \mathbb{R}^{n \times n}$. Here $Q^T Q = I$ and R is upper triangular. Let's partition these matrices:

$$A = \left(A_0 \mid a_1 \mid A_2 \right), \quad Q = \left(Q_0 \mid q_1 \mid Q_2 \right), \quad \text{and} \quad \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array} \right),$$

where $A_0, Q_0 \in \mathbb{R}^{m \times k}$ and $R_{00} \in \mathbb{R}^{k \times k}$. Now, $A = QR$ means that

$$\left(A_0 \mid a_1 \mid A_2 \right) = \left(Q_0 \mid q_1 \mid Q_2 \right) \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline 0 & \rho_{11} & r_{12}^T \\ \hline 0 & 0 & R_{22} \end{array} \right)$$

so that

$$\left(A_0 \mid a_1 \mid A_2 \right) = \left(Q_0 R_{00} \mid Q_0 r_{01} + \rho_{11} q_1 \mid Q_0 R_{02} + q_1 r_{12}^T + Q_2 R_{22} \right).$$

Now, assume that Q_0 and R_{00} have already been computed so that $A_0 = Q_0 R_{00}$. Let's focus on how to compute the next column of Q , q_1 , and the next column of R , $\left(\begin{array}{c} r_{01} \\ \rho_{11} \end{array} \right)$:

$$a_1 = Q_0 r_{01} + \rho_{11} q_1$$

implies that

$$Q_0^T a_1 = Q_0^T (Q_0 r_{01} + \rho_{11} q_1) = \underbrace{Q_0^T Q_0}_I r_{01} + \rho_{11} \underbrace{Q_0^T q_1}_0 = r_{01},$$

since $Q_0^T Q_0 = I$ (the columns of Q_0 are orthonormal) and $Q_0^T q_1 = 0$ (q_1 is orthogonal to all the columns of Q_0). So, we can compute r_{01} as

$$r_{01} := Q_0^T a_1.$$

Now we can compute a_1^\perp , the component of a_1 orthogonal to the columns of Q_0 :

$$\begin{aligned} a_1^\perp &:= a_1 - Q_0 r_{01} \\ &= a_1 - Q_0 Q_0^T a_1 \\ &= (I - Q_0 Q_0^T) a_1, \text{ the component of } a_1 \text{ orthogonal to } C(Q_0). \end{aligned}$$

Rearranging $a_1 = Q_0 r_{01} + \rho_{11} q_1$ yields $\rho_{11} q_1 = a_1 - Q_0 r_{01} = a_1^\perp$. Now, q_1 is simply the vector of length *one* in the direction of a_1^\perp . Hence we can choose

$$\begin{aligned} \rho_{11} &:= \|a_1^\perp\|_2 \\ q_1 &:= a_1^\perp / \rho_{11}. \end{aligned}$$

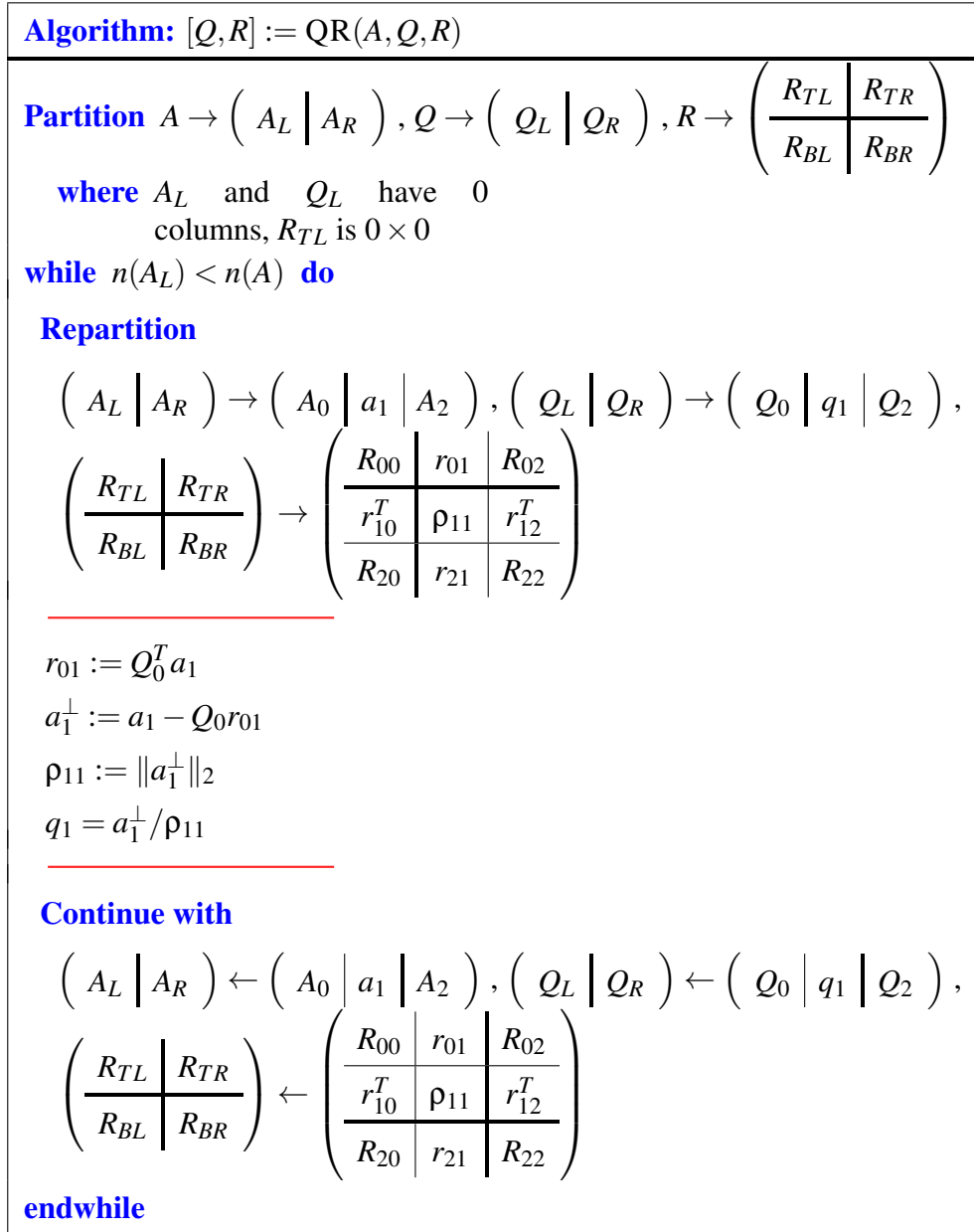


Figure 11.1: QR factorization via Gram-Schmidt orthogonalization.

All of these observations are summarized in the algorithm in Figure 11.1

Homework 11.3.7.1 Implement the algorithm for computing the QR factorization of a matrix in Figure 11.1

```
[ Q_out, R_out ] = QR_unb( A, Q, R )
```

where A and Q are $m \times n$ matrices and R is an $n \times n$ matrix. You will want to use the routines `laff_gemv`, `laff_norm`, and `laff_invscl`. (Alternatively, use native MATLAB operations.) Store the routine in

```
LAFFSpring2015 -> Programming -> Week11 -> QR_unb.m
```

Test the routine with

```
A = [ 1 -1 2
      2 1 -3
     -1 3 2
      0 -2 -1 ];
```

```
Q = zeros( 4, 3 );
R = zeros( 3, 3 );
[ Q_out, R_out ] = QR_unb( A, Q, R );
```

Next, see if $A = QR$:

```
A - Q_out * R_out
```

This should equal, approximately, the zero matrix. Check if Q has mutually orthogonal columns:

```
Q_out' * Q_out
```

This should equal, approximately, the identity matrix.

Finally, repeat the above, but with matrix

```
epsilon = 1e-8
```

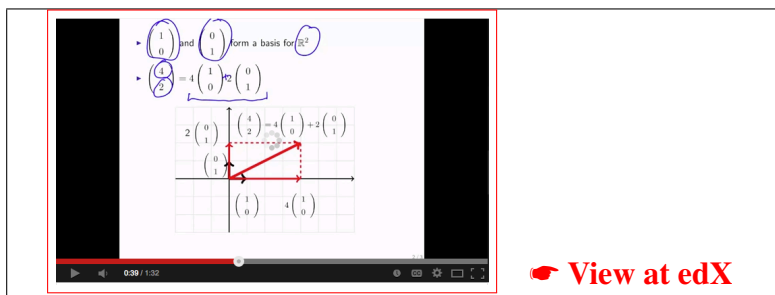
```
A = [ 1 1 1
      epsilon 0 0
      0 epsilon 0
      0 0 epsilon ]
```

```
Q = zeros( 4, 3 );
R = zeros( 3, 3 );
[ Q_out, R_out ] = QR_unb( A, Q, R );
```

Again, check if $A = QR$ and if Q has mutually orthogonal columns. To understand what went wrong, you may want to read Robert's notes for his graduate class. For details, see the enrichment for this week.

11.4 Change of Basis

11.4.1 The Unit Basis Vectors, One More Time



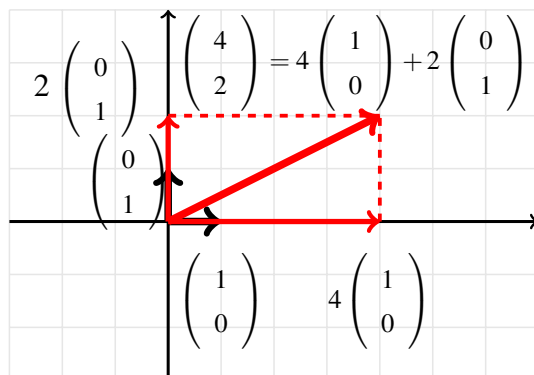
Once again, recall the unit basis vectors in \mathbb{R}^2 :

$$e_0 = \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad e_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

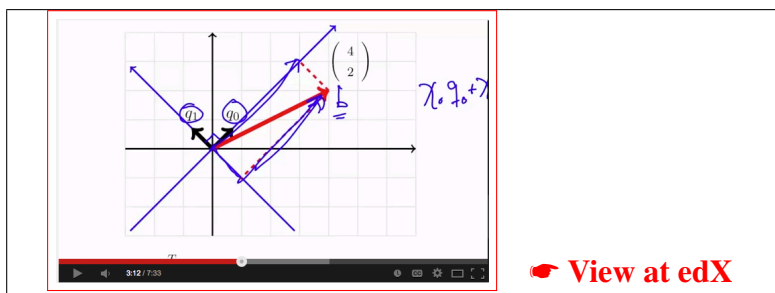
Now,

$$\begin{pmatrix} 4 \\ 2 \end{pmatrix} = 4 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 0 \\ 1 \end{pmatrix}$$

by which we illustrate the fact that $\begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\begin{pmatrix} 0 \\ 1 \end{pmatrix}$ form a basis for \mathbb{R}^2 and the vector $\begin{pmatrix} 4 \\ 2 \end{pmatrix}$ can then be written as a linear combination of these basis vectors, with coefficients 4 and 2. We can illustrate this with



11.4.2 Change of Basis



Similar to the example from the last unit, we could have created an alternate coordinate system with basis vectors

$$q_0 = \frac{\sqrt{2}}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

What are the coefficients for the linear combination of these two vectors (q_0 and q_1) that produce the vector $\begin{pmatrix} 4 \\ 2 \end{pmatrix}$? First let's look at a few exercises demonstrating how special these vectors that we've chosen are.

Homework 11.4.2.1 The vectors

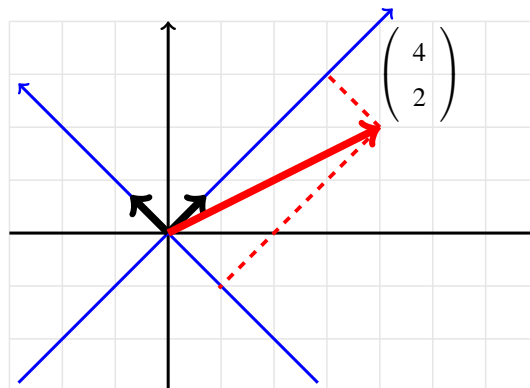
$$q_0 = \frac{\sqrt{2}}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} = \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}, \quad q_1 = \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.$$

are mutually orthonormal.

True/False

Homework 11.4.2.2 If $Q \in \mathbb{R}^{n \times n}$ has mutually orthonormal columns then which of the following are true:

- | | |
|-------------------|------------|
| 1. $Q^T Q = I$ | True/False |
| 2. $Q Q^T = I$ | True/False |
| 3. $Q Q^{-1} = I$ | True/False |
| 4. $Q^{-1} = Q^T$ | True/False |



What we would like to determine are the coefficients χ_0 and χ_1 such that

$$\chi_0 \frac{\sqrt{2}}{2} \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \chi_1 \frac{\sqrt{2}}{2} \begin{pmatrix} -1 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}.$$

This can be alternatively written as

$$\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

In Homework [11.4.2.1](#) we noticed that

$$\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

and hence

$$\underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q \begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

I

or, equivalently,

$$\begin{pmatrix} \chi_0 \\ \chi_1 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \\ -4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 3\sqrt{2} \\ -\sqrt{2} \end{pmatrix}$$

so that

$$3\sqrt{2} \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} - \sqrt{2} \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix}.$$

In other words: In the new basis, the coefficients are $3\sqrt{2}$ and $-\sqrt{2}$.

Another way of thinking of the above discussion is that

$$4 \begin{pmatrix} 1 \\ 0 \end{pmatrix} + 2 \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 4 \\ 2 \end{pmatrix}$$

$$\begin{aligned}
&= \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_{Q^T} \begin{pmatrix} 4 \\ 2 \end{pmatrix} = \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q \begin{pmatrix} 4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \\ -4\frac{\sqrt{2}}{2} + 2\frac{\sqrt{2}}{2} \end{pmatrix} \\
&= \underbrace{\begin{pmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{pmatrix}}_Q \begin{pmatrix} 3\sqrt{2} \\ -\sqrt{2} \end{pmatrix} = 3\sqrt{2} \begin{pmatrix} \frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix} - \sqrt{2} \begin{pmatrix} -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} \end{pmatrix}.
\end{aligned}$$

This last way of looking at the problem suggest a way of finding the coefficients for any basis, $a_0, a_1, \dots, a_{n-1} \in \mathbb{R}^n$. Let $b \in \mathbb{R}^n$ and let $A = \left(a_0 \mid a_1 \mid \dots \mid a_{n-1} \right)$. Then

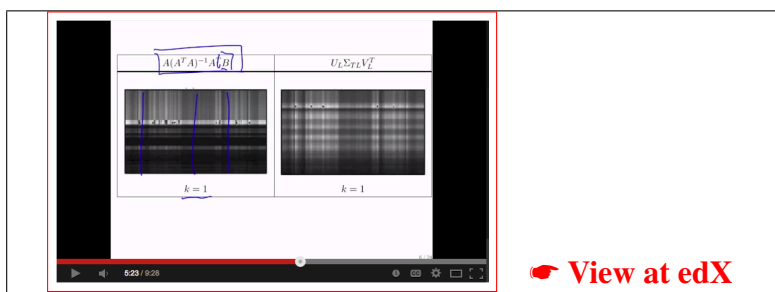
$$b = \underbrace{AA^{-1}}_I b = Ax = \chi_0 a_0 + \chi_1 a_1 + \dots + \chi_{n-1} a_{n-1}.$$

So, when the basis is changed from the unit basis vectors to the vectors a_0, a_1, \dots, a_{n-1} , the coefficients change from $\beta_0, \beta_1, \dots, \beta_{n-1}$ (the components of the vector b) to $\chi_0, \chi_1, \dots, \chi_{n-1}$ (the components of the vector x).

Obviously, instead of computing $A^{-1}b$, one can instead solve $Ax = b$.

11.5 Singular Value Decomposition

11.5.1 The Best Low Rank Approximation



Earlier this week, we showed that by taking a few columns from matrix B (which encoded the picture), and projecting onto those columns we could create a rank- k approximation, AW^T , that approximated the picture. The columns in A were chosen from the columns of B .

Now, what if we could choose the columns of A to be the *best* columns onto which to project? In other words, what if we could choose the columns of A so that the subspace spanned by them minimized the error in the approximation AW^T when we choose $W = (A^T A)^{-1} A^T B$?

The answer to how to obtain the answers the above questions go beyond the scope of an introductory undergraduate linear algebra course. But let us at least look at some of the results.

One of the most important results in linear algebra is the **Singular Value Decomposition Theorem** which says that any matrix $B \in \mathbb{R}^{m \times n}$ can be written as the product of three matrices, the Singular Value

Decomposition (SVD):

$$B = U\Sigma V^T$$

where

- $U \in \mathbb{R}^{m \times r}$ and $U^T U = I$ (U has orthonormal columns).
- $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix with positive diagonal elements that are ordered so that $\sigma_{0,0} \geq \sigma_{1,1} \geq \dots \geq \sigma_{(r-1),(r-1)} > 0$.
- $V \in \mathbb{R}^{n \times r}$ and $V^T V = I$ (V has orthonormal columns).
- r equals the rank of matrix B .

If we partition

$$U = \left(U_L \mid U_R \right), V = \left(V_L \mid V_R \right), \text{ and } \Sigma = \left(\begin{array}{c|c} \Sigma_{TL} & 0 \\ \hline 0 & \Sigma_{BR} \end{array} \right),$$

where U_L and V_L have k columns and Σ_{TL} is $k \times k$, then $U_L \Sigma_{TL} V_L^T$ is the “best” rank- k approximation to matrix B . So, the “best” rank- k approximation $B = AW^T$ is given by the choices $A = U_L$ and $W = \Sigma_{TL} V_L$.

The sequence of pictures in Figures 11.2 and 11.3 illustrate the benefits of using a rank- k update based on the SVD.

Homework 11.5.1.1 Let $B = U\Sigma V^T$ be the SVD of B , with $U \in \mathbb{R}^{m \times r}$, $\Sigma \in \mathbb{R}^{r \times r}$, and $V \in \mathbb{R}^{n \times r}$. Partition

$$U = \left(u_0 \mid u_1 \mid \dots \mid u_{r-1} \right), \quad \Sigma = \left(\begin{array}{c|c|c|c} \sigma_0 & 0 & \dots & 0 \\ \hline 0 & \sigma_1 & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \hline 0 & 0 & \dots & \sigma_{r-1} \end{array} \right), \quad V = \left(v_0 \mid v_1 \mid \dots \mid v_{r-1} \right).$$

$$U\Sigma V^T = \sigma_0 u_0 v_0^T + \sigma_1 u_1 v_1^T + \dots + \sigma_{r-1} u_{r-1} v_{r-1}^T.$$

Always/Sometimes/Never

Homework 11.5.1.2 Let $B = U\Sigma V^T$ be the SVD of B with $U \in \mathbb{R}^{m \times r}$, $\Sigma \in \mathbb{R}^{r \times r}$, and $V \in \mathbb{R}^{n \times r}$.

$$\bullet \quad \mathcal{C}(B) = \mathcal{C}(U)$$

Always/Sometimes/Never

$$\bullet \quad \mathcal{R}(B) = \mathcal{C}(V)$$

Always/Sometimes/Never

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, and $b \in \mathbb{R}^m$, we can solve $Ax \approx b$ for the “best”

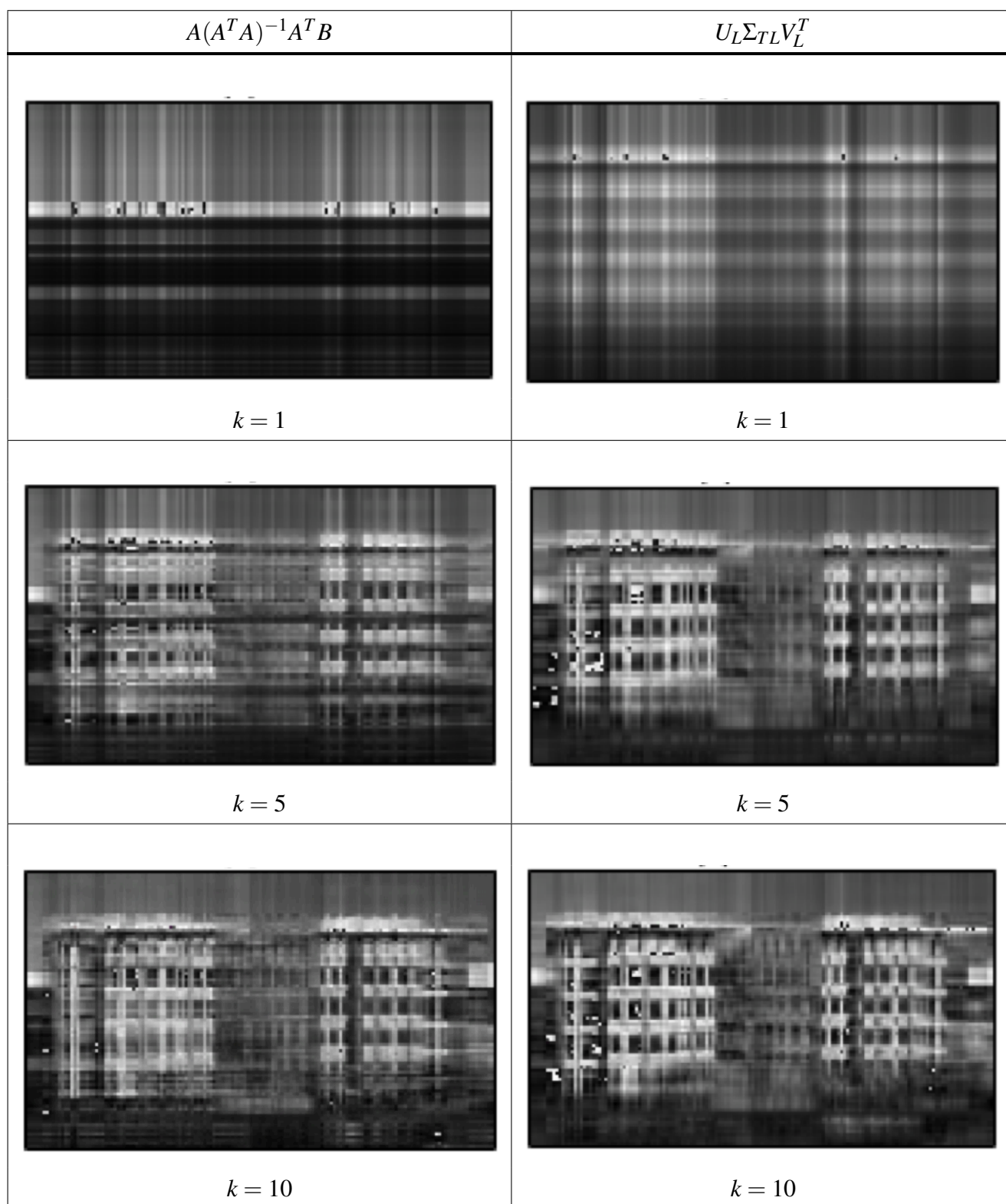


Figure 11.2: Rank-k approximation using columns from the picture versus using the SVD. (Part 1)





$A(A^T A)^{-1} A^T B$	$U_L \Sigma_{TL} V_L^T$
 <p>$k = 25$</p>	 <p>$k = 25$</p>
 <p>$k = 50$</p>	 <p>$k = 50$</p>

Figure 11.3: Rank-k approximation using columns from the picture versus using the SVD. (Continued)

solution (in the linear least-squares sense) via its SVD, $A = U\Sigma V^T$, by observing that

$$\begin{aligned}
 \hat{x} &= (A^T A)^{-1} A^T b \\
 &= ((U\Sigma V^T)^T (U\Sigma V^T))^{-1} (U\Sigma V^T)^T b \\
 &= (V\Sigma^T U^T U\Sigma V^T)^{-1} V\Sigma^T U^T b \\
 &= (V\Sigma\Sigma V^T)^{-1} V\Sigma U^T b \\
 &= ((V^T)^{-1} (\Sigma\Sigma)^{-1} V^{-1}) V\Sigma U^T b \\
 &= V\Sigma^{-1} \Sigma^{-1} \Sigma U^T b \\
 &= V\Sigma^{-1} U^T b.
 \end{aligned}$$

Hence, the “best” solution is given by

$$\hat{x} = V\Sigma^{-1}U^Tb.$$

Homework 11.5.1.3 You will now want to revisit exercise 11.2.5.2 and compare an approximation by projecting onto a few columns of the picture versus using the SVD to approximate. You can do so by executing the script `Week11/CompressPictureWithSVD.m` that you downloaded in `Week11.zip`. That script creates three figures: the first is the original picture. The second is the approximation as we discussed in Section 11.2.5. The third uses the SVD. Play with the script, changing variable `k`.

11.6 Enrichment

11.6.1 The Problem with Computing the QR Factorization

Modified Gram-Schmidt

In theory, the Gram-Schmidt process, started with a set of linearly independent vectors, yields an orthonormal basis for the span of those vectors. In practice, due to round-off error, the process can result in a set of vectors that are far from mutually orthonormal. A minor modification of the Gram-Schmidt process, known as Modified Gram-Schmidt, partially fixes this.

A more advanced treatment of Gram-Schmidt orthonormalization, including the Modified Gram-Schmidt process, can be found in Robert’s notes for his graduate class on Numerical Linear Algebra, available from <http://www.ulaff.net>.

Many linear algebra texts also treat this material.

11.6.2 QR Factorization Via Householder Transformations (Reflections)

If orthogonality is important, an alternative algorithm for computing the QR factorization is employed, based on Householder transformations (reflections). This approach resembles LU factorization with Gauss transforms, except that at each step a reflection is used to zero elements below the current diagonal.

QR factorization via Householder transformations is discussed in Robert’s notes for his graduate class on Numerical Linear Algebra, available from <http://www.ulaff.net>.

Graduate level texts on numerical linear algebra usually treat this topic, as may some more advanced undergraduate texts.

11.6.3 More on SVD

The SVD is possibly the most important topic in linear algebra.

A thorough treatment of the SVD can be found in Robert’s notes for his graduate class on Numerical Linear Algebra, available from <http://www.ulaff.net>.

Graduate level texts on numerical linear algebra usually treat this topic, as may some more advanced undergraduate texts.

11.7 Wrap Up

11.7.1 Homework

No additional homework this week.

11.7.2 Summary

Projection

Given $a, b \in \mathbb{R}^m$:

- Component of b in direction of a :

$$u = \frac{a^T b}{a^T a} a = a(a^T a)^{-1} a^T b.$$

- Matrix that projects onto $\text{Span}(\{a\})$:

$$a(a^T a)^{-1} a^T$$

- Component of b orthogonal to a :

$$w = b - \frac{a^T b}{a^T a} a = b - a(a^T a)^{-1} a^T b = (I - a(a^T a)^{-1} a^T) b.$$

- Matrix that projects onto $\text{Span}(\{a\})^\perp$:

$$I - a(a^T a)^{-1} a^T$$

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns and vector $b \in \mathbb{R}^m$:

- Component of b in $C(A)$:

$$u = A(A^T A)^{-1} A^T b.$$

- Matrix that projects onto $C(A)$:

$$A(A^T A)^{-1} A^T.$$

- Component of b in $C(A)^\perp = \mathcal{N}(A^T)$:

$$w = b - A(A^T A)^{-1} A^T b = (I - A(A^T A)^{-1} A^T) b.$$

- Matrix that projects onto $C(A)^\perp = \mathcal{N}(A^T)$:

$$(I - A(A^T A)^{-1} A^T).$$

“Best” rank- k approximation of $B \in \mathbb{R}^{m \times n}$ using the column space of $A \in \mathbb{R}^{m \times k}$ with linearly independent columns:

$$A(A^T A)^{-1} A^T B = A V^T, \quad \text{where } V^T = (A^T A)^{-1} A^T B.$$

Orthonormal vectors and spaces

Definition 11.3 Let $q_0, q_1, \dots, q_{k-1} \in \mathbb{R}^m$. Then these vectors are (mutually) orthonormal if for all $0 \leq i, j < k$:

$$q_i^T q_j = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise.} \end{cases}$$

Theorem 11.4 A matrix $Q \in \mathbb{R}^{m \times n}$ has mutually orthonormal columns if and only if $Q^T Q = I$.

Given $q, b \in \mathbb{R}^m$, with $\|q\|_2 = 1$ (q of length one):

- Component of b in direction of q :

$$u = q^T b q = q q^T b.$$

- Matrix that projects onto $\text{Span}(\{q\})$:

$$q q^T$$

- Component of b orthogonal to q :

$$w = b - q^T b q = (I - q q^T) b.$$

- Matrix that projects onto $\text{Span}(\{q\})^\perp$:

$$I - q q^T$$

Given matrix $Q \in \mathbb{R}^{m \times n}$ with mutually orthonormal columns and vector $b \in \mathbb{R}^m$:

- Component of b in $C(Q)$:

$$u = Q Q^T b.$$

- Matrix that projects onto $C(Q)$:

$$Q Q^T.$$

- Component of b in $C(Q)^\perp = \mathcal{N}(Q)$:

$$w = b - Q Q^T b = (I - Q Q^T) b.$$

- Matrix that projects onto $C(Q)^\perp = \mathcal{N}(Q)$:

$$(I - Q Q^T).$$

“Best” rank- k approximation of $B \in \mathbb{R}^{m \times n}$ using the column space of $Q \in \mathbb{R}^{m \times k}$ with mutually orthonormal columns:

$$Q Q^T B = Q V^T, \quad \text{where } V^T = Q^T B.$$

Gram-Schmidt orthogonalization

Starting with linearly independent vectors $a_0, a_1, \dots, a_{n-1} \in \mathbb{R}^m$, the following algorithm computes the mutually orthonormal vectors $q_0, q_1, \dots, q_{n-1} \in \mathbb{R}^m$ such that $\text{Span}(\{a_0, a_1, \dots, a_{n-1}\}) = \text{Span}(\{q_0, q_1, \dots, q_{n-1}\})$:

```

for  $k = 0, \dots, n-1$ 
    for  $p = 0, \dots, k-1$ 
         $\rho_{p,k} := q_p^T a_k$ 
    endfor
     $\begin{pmatrix} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{pmatrix} = \begin{pmatrix} q_0^T a_k \\ q_1^T a_k \\ \vdots \\ q_{k-1}^T a_k \end{pmatrix} = \begin{pmatrix} q_0^T \\ q_1^T \\ \vdots \\ q_{k-1}^T \end{pmatrix} a_k = \left( q_0 \mid q_1 \mid \dots \mid q_{k-1} \right)^T a_k$ 

     $\begin{matrix} a_k^\perp := a_k \\ \textbf{for } j = 0, \dots, k-1 \\ a_k^\perp := a_k^\perp - \rho_{j,k} q_j \\ \textbf{endfor} \end{matrix} \left\{ \begin{matrix} a_k^\perp = a_k - \sum_{j=0}^{k-1} \rho_{j,k} q_j = a_k - \left( q_0 \mid q_1 \mid \dots \mid q_{k-1} \right) \begin{pmatrix} \rho_{0,k} \\ \rho_{1,k} \\ \vdots \\ \rho_{k-1,k} \end{pmatrix} \end{matrix} \right.$ 

     $\begin{matrix} \rho_{k,k} := \|a_k^\perp\|_2 \\ q_k := a_k^\perp / \rho_{k,k} \end{matrix} \left\{ \begin{matrix} \text{Normalize } a_k^\perp \text{ to be of length one.} \end{matrix} \right.$ 
endfor

```

The QR factorization

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, there exists a matrix $Q \in \mathbb{R}^{m \times n}$ with mutually orthonormal columns and upper triangular matrix $R \in \mathbb{R}^{n \times n}$ such that $A = QR$.

If one partitions

$$A = \left(a_0 \mid a_1 \mid \dots \mid a_{n-1} \right), \quad Q = \left(q_0 \mid q_1 \mid \dots \mid q_{n-1} \right), \quad \text{and} \quad R = \begin{pmatrix} \rho_{0,0} & \rho_{0,1} & \dots & \rho_{0,n-1} \\ 0 & \rho_{1,1} & \dots & \rho_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \rho_{n-1,n-1} \end{pmatrix}$$

then

$$\underbrace{\left(a_0 \mid a_1 \mid \dots \mid a_{n-1} \right)}_A = \underbrace{\left(q_0 \mid q_1 \mid \dots \mid q_{n-1} \right)}_Q \underbrace{\begin{pmatrix} \rho_{0,0} & \rho_{0,1} & \dots & \rho_{0,n-1} \\ 0 & \rho_{1,1} & \dots & \rho_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \rho_{n-1,n-1} \end{pmatrix}}_R$$

and Gram-Schmidt orthogonalization (the Gram-Schmidt process) in the above algorithm computes the columns of Q and elements of R .

Solving the linear least-squares problem via the QR factorization

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, there exists a matrix $Q \in \mathbb{R}^{m \times n}$ with mutually orthonormal columns and upper triangular matrix $R \in \mathbb{R}^{n \times n}$ such that $A = QR$. The vector \hat{x} that is the best solution (in the linear least-squares sense) to $Ax \approx b$ is given by

- $\hat{x} = (A^T A)^{-1} A^T b$ (as shown in Week 10) computed by solving the normal equations

$$A^T A x = A^T b.$$

- $\hat{x} = R^{-1} Q^T b$ computed by solving

$$R x = Q^T b.$$

An algorithm for computing the QR factorization (presented in FLAME notation) is given by

Algorithm: $[Q, R] := \text{QR}(A, Q, R)$

Partition $A \rightarrow \left(A_L \mid A_R \right), Q \rightarrow \left(Q_L \mid Q_R \right), R \rightarrow \left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right)$

where A_L and Q_L have 0 columns, R_{TL} is 0×0

while $n(A_L) < n(A)$ **do**

Repartition

$\left(A_L \mid A_R \right) \rightarrow \left(A_0 \mid a_1 \mid A_2 \right), \left(Q_L \mid Q_R \right) \rightarrow \left(Q_0 \mid q_1 \mid Q_2 \right),$
 $\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \rightarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$

$$r_{01} := Q_0^T a_1$$

$$a_1^\perp := a_1 - Q_0 r_{01}$$

$$\rho_{11} := \|a_1^\perp\|_2$$

$$q_1 = a_1^\perp / \rho_{11}$$

Continue with

$\left(A_L \mid A_R \right) \leftarrow \left(A_0 \mid a_1 \mid A_2 \right), \left(Q_L \mid Q_R \right) \leftarrow \left(Q_0 \mid q_1 \mid Q_2 \right),$
 $\left(\begin{array}{c|c} R_{TL} & R_{TR} \\ \hline R_{BL} & R_{BR} \end{array} \right) \leftarrow \left(\begin{array}{c|c|c} R_{00} & r_{01} & R_{02} \\ \hline r_{10}^T & \rho_{11} & r_{12}^T \\ \hline R_{20} & r_{21} & R_{22} \end{array} \right)$

endwhile

Singular Value Decomposition

Any matrix $B \in \mathbb{R}^{m \times n}$ can be written as the product of three matrices, the Singular Value Decomposition (SVD):

$$B = U\Sigma V^T$$

where

- $U \in \mathbb{R}^{m \times r}$ and $U^T U = I$ (U has orthonormal columns).
- $\Sigma \in \mathbb{R}^{r \times r}$ is a diagonal matrix with positive diagonal elements that are ordered so that $\sigma_{0,0} \geq \sigma_{1,1} \geq \dots \geq \sigma_{(r-1),(r-1)} > 0$.
- $V \in \mathbb{R}^{n \times r}$ and $V^T V = I$ (V has orthonormal columns).
- r equals the rank of matrix B .

If we partition

$$U = \left(\begin{array}{c|c} U_L & U_R \end{array} \right), V = \left(\begin{array}{c|c} V_L & V_R \end{array} \right), \text{ and } \Sigma = \left(\begin{array}{c|c} \Sigma_{TL} & 0 \\ \hline 0 & \Sigma_{BR} \end{array} \right),$$

where U_L and V_L have k columns and Σ_{TL} is $k \times k$, then $U_L \Sigma_{TL} V_L^T$ is the “best” rank- k approximation to matrix B . So, the “best” rank- k approximation $B = AW^T$ is given by the choices $A = U_L$ and $W = \Sigma_{TL} V_L$.

Given $A \in \mathbb{R}^{m \times n}$ with linearly independent columns, and $b \in \mathbb{R}^m$, the “best” solution to $Ax \approx b$ (in the linear least-squares sense) via its SVD, $A = U\Sigma V^T$, is given by

$$\hat{x} = V\Sigma^{-1}U^T b.$$

