Revised 8/26/08

LAB 6: Orthonormal Bases, Orthogonal Projections, and Least Squares

In this lab you will use MATLAB to study the following topics:

- Geometric aspects of vectors —*norm*, *dot product*, and *orthogonal projection* onto a line.
- The *Gram-Schmidt Algorithm* to change an independent set of vectors into an *orthonormal* set, and the associated $A = QR$ matrix factorization.
- The *orthogonal projection* of a vector onto a subspace.
- The best *approximate solution* to an inconsistent linear system $A\mathbf{x} = \mathbf{b}$.
- The method of *least squares* for fitting straight lines and parabolas to data points.

Preliminaries

Reading from Textbook: In connection with this Lab, read through Sections 6.1 to 6.6 of the text and work the suggested problems for each section.

Tcodes and Script Files: For this lab you will need the Teaching Codes

```
grams.m,  linefit.m,  lsq.m, partic.m
```

Before beginning work on the Lab questions you should copy these codes from the Teaching Codes directory on the Math Department/Course Materials/Linear Algebra 250 web page to your diskette or hard drive (see Lab 3 for more details).

You will also need the m-files `rmat.m` and `rvect.m` from Lab 2. These files should already be in your directory. If you didn't do Lab 2, get a copy of that lab assignment and create these files as indicated there.

Lab Write-up: You should open a diary file at the beginning of each MATLAB session (see Lab 1 for details). Be sure to answer all the questions in the lab assignment.

Random Seed: When you start your MATLAB session, initialize the random number generator by typing

```
rand('seed', abcd)
```

where *abcd* are the last four digits of your Student ID number. This will ensure that you generate your own particular random vectors and matrices.

BE SURE TO INCLUDE THIS LINE IN YOUR LAB WRITE-UP

The lab report that you hand in must be your own work. The following problems all use randomly generated matrices and vectors, so the matrices and vectors in your lab report will not be the same as those of other students doing the lab. Sharing of lab report files is not allowed in this course.

Question 1. Norm, Dot Product, and Orthogonal Projection onto a Line

Generate random vectors $\mathbf{u}, \mathbf{v} \in \mathcal{R}^2$ by `u = rvect(2), v = rvect(2)`. Use these vectors in the following.

(a) The *norm* $||\,\mathbf{u}\,||$ of a vector is calculated by the MATLAB command `norm(u)`. The *triangle inequality* asserts that $||\,\mathbf{u}+\mathbf{v}\,|| \le ||\,\mathbf{u}\,|| + ||\,\mathbf{v}\,||$. Use MATLAB to verify that your vectors satisfy this inequality.

(b) The *dot product* $\mathbf{u} \cdot \mathbf{v}$ is calculated in MATLAB by `u'*v` when $\mathbf{u}$ and $\mathbf{v}$ are column vectors of the same size. The absolute value $|t|$ of a number t is calculated in MATLAB by `abs(t)`. The *Cauchy-Schwarz inequality* asserts that $|\mathbf{u} \cdot \mathbf{v}| \le ||\,\mathbf{u}\,||\,||\,\mathbf{v}\,||$. Use MATLAB to verify that your vectors satisfy this inequality.

(c) The *orthogonal projection* of the vector $\mathbf{u}$ onto the line $\mathcal{L}$ (one-dimensional subspace) spanned by the vector $\mathbf{v}$ is

$$\mathbf{w} = \frac{\mathbf{u} \cdot \mathbf{v}}{||\,\mathbf{v}\,||^2}\,\mathbf{v}$$

(see Figure 6.3 on page 366 of the text). Use MATLAB to calculate $\mathbf{w}$ for your vectors. Two vectors are *orthogonal* if their dot product is zero. Verify by MATLAB that the vector $\mathbf{z} = \mathbf{u} - \mathbf{w}$ is orthogonal to $\mathbf{v}$. (If the dot product is not exactly zero but is a very small number of size 10^{-13} for example, then the vectors are considered orthogonal for numerical purposes.)

(d) The formula for $\mathbf{w}$ in **(c)** can also be written as a matrix-vector product. Use MATLAB to obtain the matrix

```
P = v*inv(v'*v)*v'
```

(note carefully the punctuation and the order of the factors in this formula).

(i) Explain why P is a 2×2 matrix, although $\mathbf{v}$ is a vector.

Calculate by MATLAB that $P\mathbf{u}$ is the vector $\mathbf{w}$ for your $\mathbf{u}$ and $\mathbf{v}$.

(ii) Write out (by hand) an algebraic proof that $\mathbf{u} - P\mathbf{u}$ must be orthogonal to $\mathbf{v}$. Use general vectors $\mathbf{u}$ and $\mathbf{v}$ with symbolic entries, not the particular random numerical vectors that you generated.

Question 2. Gram-Schmidt Orthogonalization and QR Factorization

Generate three random vectors in $\mathcal{R}^3$ by

```
u1 = rvect(3), u2 = rvect(3), u3 = rvect(3)
```

Use these vectors in the following.

(a) Since the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$ are chosen at random, it is very likely that they are linearly independent but not mutually orthogonal. To check this graphically with MATLAB, generate a line plotting parameter r and open a graphics window by the commands

```
r = 0:0.05:1; hold on
```

(be careful with the punctuation). Plot the three vectors in the graphics window as red, green, and blue dotted lines by the commands

```
plot3(r*u1(1),r*u1(2),r*u1(3), 'r:')
plot3(r*u2(1),r*u2(2),r*u2(3), 'g:')
plot3(r*u3(1),r*u3(2),r*u3(3), 'b:')
```

(use the up-arrow key $\uparrow$ to save typing). Using the Rotate 3D command, determine visually whether the vectors are independent, and whether they are mutually orthogonal. Insert a comment into your diary file.

(b) Now use the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$ to obtain an orthogonal basis for $\mathcal{R}^3$, following the Gram-Schmidt algorithm (see the proof of Theorem 6.6 on page 323 of the text). Set `v1 = u1`. Define the projection P_1 onto the span of $\mathbf{v}_1$ as in Question 1(d), and obtain $\mathbf{v}_2$ by removing the component of $\mathbf{u}_2$ in the direction $\mathbf{v}_1$:

```
P1 = v1*inv(v1'*v1)*v1', v2 = u2 - P1*u2
```

Calculate the dot product to check that the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ are mutually orthogonal (within a negligible numerical error). Also add $\mathbf{v}_2$ to your graphics window as a dashed-dotted green line by

```
plot3(r*v2(1),r*v2(2),r*v2(3), 'g-.')
```

Using the Rotate 3D command, rotate the frame to see that $\mathbf{v}_1$ and $\mathbf{v}_2$ are orthogonal.

Next obtain P_2 as the projection onto the span of $\mathbf{v}_2$ and obtain $\mathbf{v}_3$ by removing the components of $\mathbf{u}_3$ in the directions of $\mathbf{v}_1$ and $\mathbf{v}_2$:

```
P2 = v2*inv(v2'*v2)*v2', v3 = u3 - P1*u3 - P2*u3
```

Calculate dot products by MATLAB to check that $\mathbf{v}_3$ is orthogonal to the vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ (within a negligible numerical error). Add $\mathbf{v}_3$ to your plot as a dashed-dotted blue line by

```
plot3(r*v3(1),r*v3(2),r*v3(3), 'b-.')
```

Make a print-out of the graph with the vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3$ and $\mathbf{v}_2, \mathbf{v}_3$. Use Rotate 3D to obtain a good alignment of the graph that shows orthogonality in perspective.

(c) The last step in the Gram-Schmidt algorithm is to rescale the vectors $\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3$ to obtain an *orthonormal basis* for $\mathcal{R}^3$:

```
w1 = v1/norm(v1), w2 = v2/norm(v2), w3 = v3/norm(v3)
```

Check (by MATLAB) that $||\mathbf{w}_i|| = 1$ for $i = 1, 2, 3$. Close the graphics window and then plot $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$ in a new graphics window by the commands

```
hold on
plot3(r*w1(1),r*w1(2),r*w1(3), 'r+')
plot3(r*w2(1),r*w2(2),r*w2(3), 'g+')
plot3(r*w3(1),r*w3(2),r*w3(3), 'b+')
```

(this will plot the vectors as red, green, and blue ++++). Make a print-out of the graph, using Rotate 3D as necessary to get a good picture of this orthonormal frame.

(d) The Gram-Schmidt algorithm gives the QR factorization of a matrix (see the boxed statement on page 382 of the text). To illustrate this using MATLAB, set

```
Q = [w1, w2, w3]
```

(i) What is the matrix $Q' * Q$?

(ii) What is the *inverse matrix* Q^{-1}?

Answer both these questions without MATLAB and then check with a MATLAB calculation.

Set

```
A = [u1, u2, u3], R = Q'*A
```

Verify by MATLAB that $A = Q * R$.

(iii) Use the formulas in part **(b)** and the fact that each column of QR is a linear combination of the columns of Q to explain (with a hand calculation) why R is an upper-triangular matrix

Question 3. Orthogonal Projections

Generate three random vectors $\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3 \in \mathcal{R}^5$ and the matrix A with these vectors as columns:

```
u1 = rvect(5); u2 = rvect(5); u3 = rvect(5); A = [u1, u2, u3]
```

Use these vectors and this matrix in the following.

(a) Let $W = \text{Col}(A)$ be the subspace of $\mathcal{R}^5$ spanned by $\{\mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_3\}$. What is $\dim(W)$? Justify your answer by an appropriate rank calculation using MATLAB.

The teaching code `grams.m` carries out the steps of the Gram-Schmidt algorithm, just as you did step-by-step in Question 2. Calculate

```
Q = grams(A); w1 = Q(:,1), w2 = Q(:,2), w3 = Q(:,3)
```

by MATLAB. Calculate `Q'*Q` by MATLAB. Explain why the matrix obtained for `Q'*Q` shows that $\{\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3\}$ is an orthonormal set.

(b) The orthogonal projection P from $\mathbf{R}^5$ onto the subspace W is given by the 5×5 matrix

```
P = w1*w1' + w2*w2' + w3*w3'
```

(note that `w1` is a 5×1 column vector; hence `w1'` is a 1×5 row vector and `w1*w1'` is a 5×5 matrix). If $\mathbf{v} \in \mathcal{R}^5$ then

$$P\mathbf{v} = (\mathbf{w}_1 \cdot \mathbf{v})\mathbf{w}_1 + (\mathbf{w}_2 \cdot \mathbf{v})\mathbf{w}_2 + (\mathbf{w}_3 \cdot \mathbf{v})\mathbf{w}_3$$

(see the boxed formula on page 376 of the text). Verify by MATLAB that $P - P^{\mathrm{T}} = 0$ and $P^2 - P = 0$ (these are the properties that characterize an orthogonal projection matrix). Verify by MATLAB that $P\mathbf{u}_1 = \mathbf{u}_1$, $P\mathbf{u}_2 = \mathbf{u}_2$, and $P\mathbf{u}_3 = \mathbf{u}_3$, which confirms that P is the projection onto W.

(c) Orthogonal Decomposition $\mathbf{v} = \mathbf{w} + \mathbf{z}$: Generate another random vector `v = rvect(5)`. Set `w = P*v, z = v - w`. Verify by MATLAB that `P*w = w` and `P*z = 0`. This shows that $\mathbf{w}$ is in the subspace W and that $\mathbf{z}$ is the component of $\mathbf{v}$ perpendicular W (see Figure 6.9 on page 388 of the text).

(d) The projection matrix P onto the subspace W can be calculated directly from the matrix A, without first orthogonalizing the columns of A. Define

```
PW = A*inv(A'*A)*A'
```

(see Theorem 6.8 on page 395 of the text). Check by MATLAB that `norm(PW - P)` is zero (up to negligible numerical error).

Question 4. Approximate Solution to Inconsistent Linear System

For this question use the 5×3 matrix A and the vector $\mathbf{v} \in \mathcal{R}^5$ from Question 3.

(a) Approximate Solution to $A\mathbf{x} = \mathbf{v}$: Let $\mathbf{v} = \mathbf{w} + \mathbf{z}$ be the orthogonal decomposition from Question 3(c). Show that

(i) The equation $A\mathbf{x} = \mathbf{v}$ is *inconsistent* (has no solutions).

(ii) The equation $A\mathbf{x} = \mathbf{w}$ is *consistent.*

(Calculate the rank of the augmented matrix in each case).

Now solve the equation (ii) by

```
xls = inv(A'*A)*A'*v
```

(see page 404 of the text). Check by MATLAB that `A*xls = w` (up to negligible numerical error). Denote the vector `xls` by $\bar{\mathbf{x}}$. It is called the *least squares* approximate solution to the inconsistent equation $A\mathbf{x} = \mathbf{v}$.

(b) Closest Vector Property: Generate a random vector `y = rvect(3)` and verify by MATLAB that `P*A*y = A*y`. This shows that $A\mathbf{y} \in W$. Since $A\bar{\mathbf{x}} = \mathbf{w}$, the vector $A\bar{\mathbf{x}}$ is the vector in W that is closest to $\mathbf{v}$. Thus the function $||A\mathbf{y} - \mathbf{v}||$ is *minimized* by choosing $\mathbf{y} = \bar{\mathbf{x}}$. This is the *closest vector* property (see page 397 of the text), which is also called the *least squares* property. To illustrate this, use the MATLAB `norm` function to verify that $||A\bar{\mathbf{x}} - \mathbf{v}|| < ||A\mathbf{y} - \mathbf{v}||$.

Question 5. Fitting Curves to Data Points

Read Section 6.4 of the text and work the suggested exercises for that Section.

(a) Generating and Plotting Linear Data: Define a column vector of ten equally-spaced t values

```
t = [1:10]'
```

Now generate a column vector of the corresponding values of the linear function $y = 4 + t$ and plot it by

```
y = 4 + t; linefit(t,y)
```

(Note the semicolon that suppresses the display of the y data). MATLAB should open a new window in which this line is plotted. Notice that the line passes exactly through each data point, and the "best fit" equation is the given function $4 + t$. Print this graph and include the printed copy in your lab write-up.

(b) Linear Data with Random Noise: Now add some random noise to each y data value in part **(a)**:

```
y = 2 + 4*rand(10,1) + t; linefit(t,y)
```

These random data points don't lie on the line from part **(a)**, even though they show the same general trend. The line that is plotted is the *best fitting* line; it minimizes the *mean square error* between the y coordinates on the line and the data values.

(i) How many data points are above the best-fitting line?

(ii) How many data points are below the best-fitting line?

The equation of the best line fitting the random data points is displayed above the graph. Label and print the figure. Include the printed copy in your lab write-up.

(c) Fitting a Parabola to Data: This part refers to Exercise #38 on page 410 of the text. Use MATLAB to define vectors $\mathbf{t}, \mathbf{y} \in \mathcal{R}^6$ using the data given in Exercise #38. Then define a 6×3 matrix C whose first column has all ones, the second column is $\mathbf{t}$, and the third column has the squares of the entries in $\mathbf{t}$. You can get the third column by the command $\mathbf{t}.\hat{}\,2$ (notice the period before the exponent). Use the method of Example 2 on page 405 of the text to calculate the vector $\mathbf{x} \in \mathcal{R}^3$ whose components a, b, c are the coefficients in the best-fitting parabola. Then plot the data points and the least-squares parabola fitting these points by the commands

```
figure; plot(t,y,'*'); hold on
s = [0:0.1:30]; u = ones(301,1); A = [u  s'  (s.^2)'];
plot(s, A*x)
```

You should get a figure with the six data points indicated by $*$ and a smooth parabola passing very close to each data point (as in Figure 6.17 on page 406 of the text). Label and print out the figure. Calculate the *relative least-squares error*

```
norm(y - C*x)/norm(y)
```

This should be less than 0.01 (a one-percent error).

Final Editing of Lab Write-up: After you have worked through all the parts of the lab assignment, you will need to edit your diary file. Remove all errors and other material that is not directly related to the questions. Your write-up should only contain the required MATLAB calculations and the answers to the questions. Preview the document before printing and remove unnecessary page breaks and blank space.