Math 521 Homework 3

Due on Thursday, March 22, 2018

Kristin Holmbeck

Contents

List of Figures

Theory

1. Gradient of Inner Product

Show that $\nabla_{\boldsymbol{v}}(\boldsymbol{v},\boldsymbol{v}) = 2\boldsymbol{v}$ and that for a symmetric matrix C, $\nabla_{\boldsymbol{v}}(\boldsymbol{v},C\boldsymbol{v}) = 2C\boldsymbol{v}$.

Assume $\boldsymbol{v} \in \mathbb{R}^n$.

$$
\begin{aligned}
\nabla_{\boldsymbol{v}}(\boldsymbol{v},\boldsymbol{v}) &= \nabla_{\boldsymbol{v}}\boldsymbol{v}^T\boldsymbol{v} \\
&= \nabla_{\boldsymbol{v}}\left(v_1^2 + v_2^2 + \cdots + v_n^2\right) \\
&= \left(\frac{\partial\left(v_1^2 + v_2^2 + \cdots + v_n^2\right)}{\partial v_1}, \ldots, \frac{\partial\left(v_1^2 + v_2^2 + \cdots + v_n^2\right)}{\partial v_n},\right) \\
&= (2v_1, 2v_2, \ldots, 2v_n) \\
&= 2\boldsymbol{v}
\end{aligned}
$$

Next, show $\nabla_{\boldsymbol{v}}(\boldsymbol{v},C\boldsymbol{v}) = 2C\boldsymbol{v}$:

$$
\begin{aligned}
\text{Let } \boldsymbol{v} &= \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \\
\text{and } C &= \begin{bmatrix} c_{11} & c_{12} & \cdots & c_{1n} \\ c_{21} & c_{22} & \cdots & c_{2n} \\ \vdots & & & \vdots \\ c_{n1} & \cdots & & c_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & c_{21} & \cdots & c_{n1} \\ c_{12} & c_{22} & \cdots & c_{n2} \\ \vdots & & & \vdots \\ c_{1n} & \cdots & & c_{nn} \end{bmatrix} \\
&= \left[\vec{c}_1 \cdots \vec{c}_n\right] \\
\text{then } \boldsymbol{v}^T C\boldsymbol{v} &= \begin{bmatrix} v_1 & \cdots & v_n \end{bmatrix} \begin{bmatrix} \vec{c}_1 \cdot \boldsymbol{v} \\ \vdots \\ \vec{c}_n \cdot \boldsymbol{v} \end{bmatrix} = v_1\vec{c}_1 \cdot \boldsymbol{v} + \cdots + v_n\vec{c}_n \cdot \boldsymbol{v} \\
&= v_1(c_{11}v_1 + c_{12}v_2 + \cdots + c_{1n}v_n) + \cdots + v_n(c_{n1}v_1 + c_{n2}v_2 + \cdots + c_{nn}v_n) \\
\text{So } \nabla_{\boldsymbol{v}}(\boldsymbol{v},C\boldsymbol{v}) &= \nabla_{\boldsymbol{v}}\boldsymbol{v}^T C\boldsymbol{v} \\
\nabla_{\boldsymbol{v}}\boldsymbol{v}^T C\boldsymbol{v} &= \begin{bmatrix} \partial v_1(\boldsymbol{v}^T C\boldsymbol{v}) \\ \vdots \\ \partial v_n(\boldsymbol{v}^T C\boldsymbol{v}) \end{bmatrix} \\
&= \left[\vec{c}_1 \cdot \boldsymbol{v} + \right] \\
\partial v_k(\boldsymbol{v}^T C\boldsymbol{v}) &= v1c_{1k} + v2c_{2k} + \cdots + \partial v_k\left[v_k(c_{k1}v1 + \cdots + c_{kk}v_k + \cdots + c_{kn}v_n)\right] + \cdots + v_n c_{nk} \\
\partial v_k(\boldsymbol{v}^T C\boldsymbol{v}) &= v1c_{1k} + v2c_{2k} + \cdots + \left[(c_{k1}v1 + \cdots + c_{kk}v_k + \cdots + c_{kn}v_n) + v_k c_{kk}\right] + \cdots + v_n c_{nk} \\
&= 2\vec{c}_k\boldsymbol{v} \\
\implies \nabla_{\boldsymbol{v}}(\boldsymbol{v},C\boldsymbol{v}) &= 2C\boldsymbol{v}
\end{aligned}
$$

2. Commutativity of Symmetric Matrix in the Inner Product

Show that for a symmetric matrix C, $(\phi^{(1)}, C\phi^{(2)}) = (C\phi^{(1)}, \phi^{(2)})$.

$$(x, Cy) = (Cy)^T x = y^T C^T x = y^T Cx = (Cx, y)$$
$$\implies (x, Cy) = (Cx, y)$$

3. Eigenvalues and eigenvectors

Given the data matrix

$$X = \begin{bmatrix} -2 & -1 & 1 \\ 0 & -1 & 0 \\ -1 & 1 & 2 \\ 1 & -1 & 1 \end{bmatrix}$$

compute the eigenvalues and eigenvectors of XX^T and X^TX. For $\boldsymbol{u}^{(1)}$, confirm the statement

$$\boldsymbol{u}^{(j)} = \frac{1}{\sigma_j} \Sigma_{k=1}^{P} v_k^{(j)} \boldsymbol{x}^{(k)}$$

By MATLAB, $\rho(XX^T) = \lambda(\lambda-9)(\lambda-4)(\lambda-3) = \lambda\rho(X^TX)$, i.e. the eigenvalues of XX^T are $\{0, 3, 4, 9\}$ and the eigenvalues of X^TX are $\{3, 4, 9\}$. See the heavy-lifting code later in this document. The eigenvectors of X^TX are the column vectors of the matrix:

$$\begin{bmatrix} -0.71 & 0.50 & -0.41 & -0.29 \\ 0.00 & 0.50 & -0.00 & 0.87 \\ -0.71 & -0.50 & 0.41 & 0.29 \\ -0.00 & 0.50 & 0.82 & -0.29 \end{bmatrix}$$

and the eigenvectors of XX^T are the column vectors of

$$\begin{bmatrix} -0.71 & 0.00 & -0.71 \\ 0.00 & -1.00 & 0.00 \\ 0.71 & 0.00 & -0.71 \end{bmatrix}$$

For $\boldsymbol{u}^{(1)}$,

$$stuffgoeshere$$

4. FLOP count

Assume A is $N \times P$, and that $N > P$. Then the SVD of A is $A = U\Sigma V^T$ where U is $N \times P$, Σ is $P \times P$, and V is $P \times P$.

$$A = \Sigma V^T = \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_P \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{1P} \\ \vdots & \ddots & \vdots \\ v_{P1} & \cdots & v_{PP} \end{bmatrix}^T$$

$$= \begin{bmatrix} \sigma_1 & & 0 \\ & \ddots & \\ 0 & & \sigma_P \\ 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} v_{11} & \cdots & v_{P1} \\ \vdots & \ddots & \vdots \\ v_{1P} & \cdots & v_{PP} \end{bmatrix}$$

$$= \begin{bmatrix} \sigma_1 \boldsymbol{v}^{(1)^T} \\ \sigma_2 \boldsymbol{v}^{(2)^T} \\ \vdots \\ \sigma_P \boldsymbol{v}^{(P)^T} \end{bmatrix} = \begin{bmatrix} P \text{ scalar multiplies} \\ P \text{ scalar multiplies} \\ \vdots \\ P \text{ scalar multiplies} \end{bmatrix} = P^2 \text{ operations}$$

On the other hand,

$$A = U^T X = \begin{bmatrix} u_{11} & \cdots & u_{N1} \\ \vdots & \ddots & \vdots \\ u_{1P} & \cdots & u_{NP} \end{bmatrix}^T \begin{bmatrix} x_{11} & \cdots & x_{N1} \\ \vdots & \ddots & \vdots \\ x_{1P} & \cdots & x_{NP} \end{bmatrix}$$

$$= \begin{bmatrix} u_{11} & \cdots & u_{P1} \\ \vdots & \ddots & \vdots \\ u_{1N} & \cdots & u_{PN} \end{bmatrix} \begin{bmatrix} x_{11} & \cdots & x_{N1} \\ \vdots & \ddots & \vdots \\ x_{1P} & \cdots & x_{NP} \end{bmatrix}$$

$$= \begin{bmatrix} P \text{ multiplies, } P-1 \text{ additions} & \cdots & P \text{ multiplies, } P-1 \text{ additions} \\ \vdots & \ddots & \vdots \\ P \text{ multiplies, } P-1 \text{ additions} & \cdots & P \text{ multiplies, } P-1 \text{ additions} \end{bmatrix}_{N \times N}$$

$$= \begin{bmatrix} 2P-1 \text{ operations} & \cdots & 2P-1 \text{ operations} \\ \vdots & \ddots & \vdots \\ 2P-1 \text{ operations} & \cdots & 2P-1 \text{ operations} \end{bmatrix}_{N \times N}$$

$$= (2P-1)N^2 \text{ operations}$$

Computing

The computing assignment is to apply the *snapshot* method to a collection of high-resolution files. We will briefly discuss the background being the method, the implementation, and provide results on a test data set.

Suppose we have a set of P $N \times N$ matrices where $P << N$. The KL expansion as discussed in class gives rise to a construction of an optimal basis for a set of vectors $\{\boldsymbol{x}^{(\mu)}\}_{\mu=1}^{P}$ characterized by:

$$C\phi^{(i)} = \lambda_i \phi^{(i)} \tag{1}$$

where $C = \frac{1}{P}\Sigma_{\mu=1}^{P}(x^{(\mu)} - \langle x\rangle)(x^{(\mu)} - \langle x\rangle)^T$ is the ensemble average covariance matrix
and $\langle x\rangle = \frac{1}{P}\Sigma_{\mu=1}^{P}x^{(\mu)}$ is the ensemble average

Notice that C is $N \times N$. When N becomes large, it is not feasible to solve this problem directly. If C is nonsingular, we can reduce (without approximation) the problem from Equation 1 into a $P \times P$ problem. This is known as the *snapshot* method.

The test data set in question involves a fixed camera in a room with a person facing the camera and moving in the foreground. The ensemble average of these data is shown in Figure 1.

Figure 1: Ensemble Average of data set

Next, we display one of the mean-subtracted images, that is, the data set minus the ensemble average (Figure 2).

Figure 2: One mean-subtracted image

Additionally, several eigen-images are shown below:

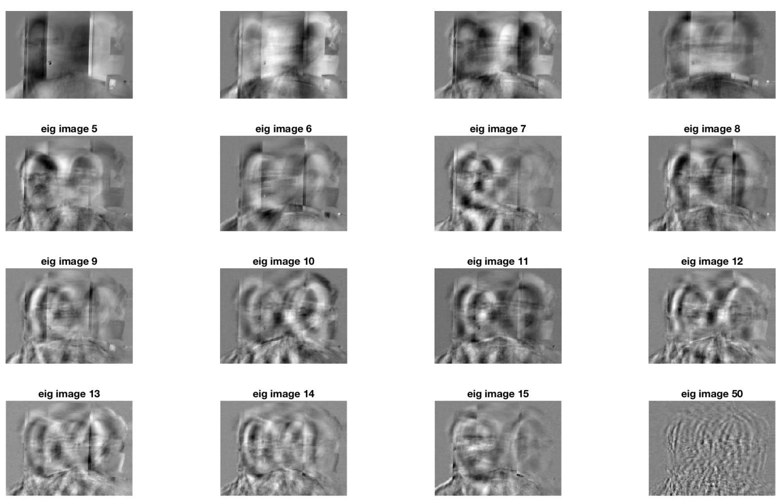

Figure 3: Some eigen-images

Next, we show the cumulative energy (as defined in a previous homework) of the mean-subtracted data set (Figure 4).

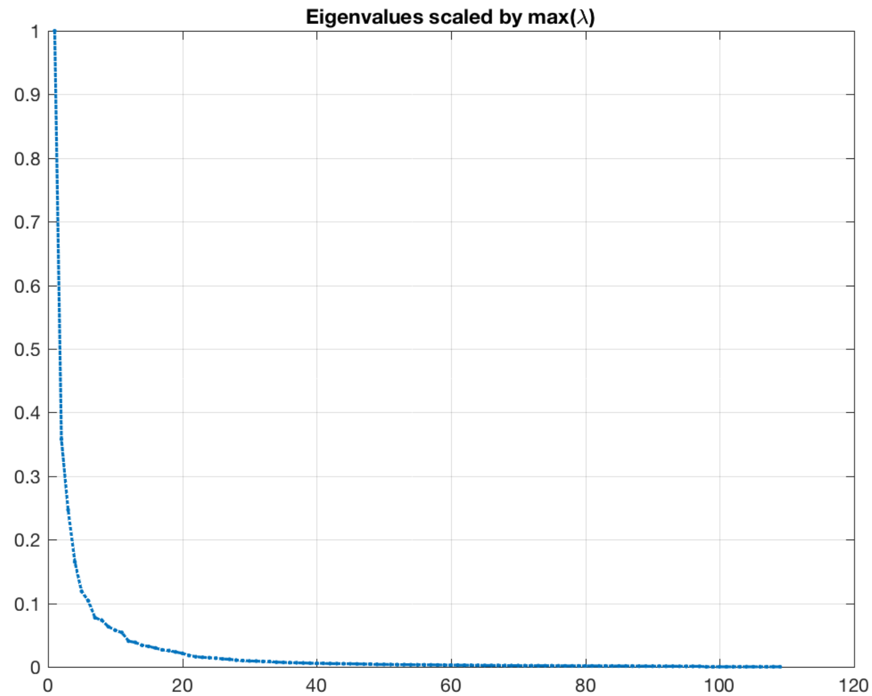

Figure 4: Scaled singular values

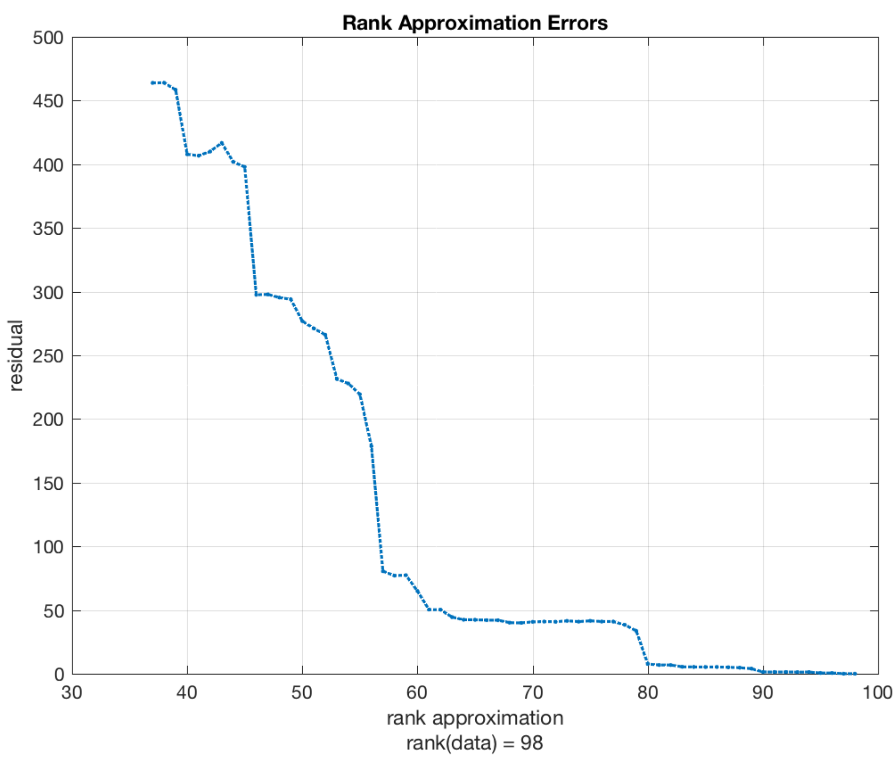

Figure 5: Rank Approximation Errors (rank $\geq$ 40)

Code

Eigenvalues and Vectors

```
clear;

X = [-2  -1  1
      0  -1  0
     -1   1  2
      1  -1  1];

[U1,S1,V1] = svd(X*X', 0);
[U2,S2,V2] = svd(X'*X, 0);
[Ut,St,Vt] = svd(X, 0);

ATA = X'*X;
AAT = X*X';

% do the SVD "by hand"

E1 = sort(eig(ATA), 'descend');
E2 = sort(eig(AAT), 'descend');

I3 = eye(3);
I4 = eye(4);

for ii = 1:4
    V_AAT{ii} = null(AAT - E2(ii)*I4);
end
V_AAT = [V_AAT{:}];

for ii = 1:3
    V_ATA{ii} = null(ATA - E1(ii)*I3);
end
V_ATA = [V_ATA{:}];

% Confirming the statement for u^{(1)}:

k = 1;
u1 = 0;
for ii = 1:size(X,2)
    u1 = u1 + X(:,ii)*Vt(ii,k);
end
% u1 = u1 / sqrt(S1(1));

return

disp_for_latex( V_AAT );
disp_for_latex( V_ATA );
```

References

[1] Chang, Jen-Mei. *Matrix Methods for Geometric Data Analysis and Recognition*. 2014.