APM120, Homework #6

Applied Linear Algebra and Big Data

Last updated: Thursday 23rd February, 2023, 17:41

Assigned March 7, due March 21, 1:00 pm, via gradescope, **in pdf $\leq$ 20Mb, $\leq$ 30 pages.**

Rank-deficient $\mathsf{A}^T\mathsf{A}$*, over-determined systems and QR, PCA/MCA*

Show all steps in all calculations explicitly. Attach code used, well documented, and relevant plots and Matlab/python output, attaching code and figures *immediately following* the relevant question solution. A code printout is not a substitute for complete solutions, your solution should stand alone without the Matlab/python code or output. See needed python preliminaries at end of this HW[1]. It is fine to use Matlab/python unless a hand calculation (using only a simple calculator) is required *in orange*. For all questions, make sure you show all steps as if you are doing the problem by hand, and do not use library functions unless explicitly allowed in the question. Make sure you can do all calculations using no more than a hand-held calculator. See the end-note for guidelines and examples of hand-calculations.[2]

1. **Rank-deficient** $\mathsf{A}^T\mathsf{A}$:

   (A) Consider the following $\mathsf{A}\mathbf{x} = \mathbf{b}$ with more equations than unknowns,

   ```
   A=[-2, -4, 0; 4, -0, 0; 0, -8, 0; -2, -20, 0];
   b=[-1; -3; -1; -3];
   ```

   (i) Try to solve using the least square approach. What do you find? Why? **(ii)** Find two different solutions $\mathbf{x}$ that both have the smallest possible norm of the residuals. **(iii)** State the assumptions needed to obtain a unique solution, explain why they are necessary and calculate it. **(iv)** Compare the solution and its norm to that of the Matlab's backslash operator or python's `x=linalg.lstsq(A,b)` and discuss.

   (B) Consider the following system with fewer equations than unknowns,

   ```
   A=[3, 6, 1; 6, 12, 2];
   b=[-5; -4];
   ```

   (i) What is the rank of A? Can an $\mathbf{x}$ that exactly solves this system be found? Why? **(ii)** State the necessary additional assumptions to obtain a unique solution, explain why they are necessary and calculate it.

2. **Over-determined systems and QR decomposition:** consider $\mathsf{A}\mathbf{x} = \mathbf{b}$ with,

   ```
   A=[-4, -1, -1; 8, -10, 0; 2, 5, -2; -7, 0, -7; -7, 1, -3];
   b=[-7; -3; -6; -2; -9];
   ```

   (**A**) Briefly derive the general least squares solution for $\mathsf{A}\mathbf{x} = \mathbf{b}$ which minimizes $\|\mathsf{A}\mathbf{x} - \mathbf{b}\|^2$ and use it to solve for $\mathbf{x}$. (**B**) Find the economical QR decomposition of A ("economical", as in SVD, means keeping only the minimum number of columns needed to reconstruct A): use the Gram-Schmidt process on the columns of A to find Q, and then, given that $\mathsf{A} = \mathsf{Q}\mathsf{R}$, multiply by the transpose (inverse) of Q to find, $\mathsf{R} = \mathsf{Q}^T\mathsf{A}$ which should be upper diagonal. (**C**) Use the QR decomposition to efficiently solve $\mathsf{A}\mathbf{x} = \mathbf{b}$ using forward/ back substitutions. (**D**) Compare the two solutions for $\mathbf{x}$ from sections A and C.

3. **PCAs via covariance matrix vs via SVD:** (**A**) Why do the two methods give the same answer? (**B**) Given the data matrix F below, calculate the PCs and expansion coefficients ("time series") using SVD, and compare them to the solution using the covariance matrix. (**C**) Most important (and the best preparation for quizzes...): describe a scenario in an application (e.g., weather at three different cities as a function of time, votes for different parties as a function of county, grades for different courses for different students, stokes for different companies on different days, etc) that fits these data.

```
F=[-0.8,  -14,  2.4, -8.4,    4, -2.4,    6,    4,  8.8;
   -7.2,   22, -8.8,   15,   -8,  6.4,   -8,   -2, -8.8;
     26,  3.2,   14, -2.4,    4, -7.2,   -6,  -14,  -18];
```

(**D**) ***** optional extra credit:** Calculate the first two PCAs for the dataset `global_temperature.mat` from https://courses.seas.harvard.edu/climate/eli/Courses/APM120/Sources/Data-for-HW/GlobalSurfaceTemperature/ (or see DropBox link on Canvas) representing the anomaly global mean temperature over the past 138 years: (i) Contour (using `contourf`) the surface temperature for the first and last years as a function of latitude (y-axis) and longitude (x-axis) using the same color range, to allow comparison. To add the coastline,

```
% Matlab:
load global_temperature.mat
% first year
TMP=permute(squeeze(T1(:,:,1)),[2,1])
contour(X,Y,TMP,[0,0])
```

```
# python:
from scipy.io import loadmat;
mat=loadmat('global_temperature.mat');
X=mat['X'];Y=mat['Y'];T1=mat['T1'];
x,y=np.meshgrid(X,Y);
plt.contour(x,y,T1[0,:,:],(0));
```

(ii) Remove the time mean from every grid point and analyze this anomaly as follows: (iii) Reshape the data into a set of column vectors, one per year; (iv) Calculate and plot the covariance of the data matrix. *Note:* the dimension of the covariance matrix should be $N \times N$, where N is the total number of spatial points. (v) Calculate the first three PCAs from the covariance and the corresponding principal component time series. (vi) Reshape the two calculated PCs back into a 2-d (latitude-longitude) structure, plot them and the corresponding time series, and interpret your results.

4. **Maximum Covariance Analysis (MCA) and multivariate PCA:** Consider the following two data sets, corresponding to the prices over N days of $M = 4$ products in one store and $L = 3$ products in a different store. Analyze possible connections between these two data sets using two approaches as follows.

```
X=[ 0.1,  2.8,  3.5,  0.5,  1.2,    4,  2.1,    0,  2.9,  3.6,  0.4,  1.1;
    3.8,  1.4,  0.5,  3.3,  2.7,    0,  1.7,  4.1,  1.1,  0.5,  3.5,  2.9;
    9.7, -1.3, -4.2,  8.1,  5.3, -5.8,  1.8,  9.9, -1.5, -4.2,  8.2,  5.5;
    9.9, -1.7, -4.3,  8.5,  5.7, -5.8,  2.3,  9.7, -1.5, -4.3,  8.3,  5.5];
Y=[-7.6, -3.6, -2.8, -6.9, -6.1, -2.2, -4.9, -7.4, -3.9, -2.7,   -7, -6.1;
   -2.1, -6.5, -7.4, -2.9, -3.8, -8.1, -5.1, -2.4, -6.3, -7.5, -2.8, -3.8;
   -2.7,   -6, -6.9, -3.1,   -4, -7.3,   -5, -2.7,   -6, -6.9, -3.1,   -4];
```

(**A**) Prepare the data for analysis by removing time-mean and normalizing by the standard deviation, showing all steps (carefully follow the normalization shown in "Multivariate PCA" section of course notes). (**B**) Combine the two data sets into a single

data matrix using Matlab: `F=[X;Y]`; python: `F=np.vstack([X,Y]);` and perform multivariate PCA ("multivariate", because the data matrix contains two data sets) using SVD of the combined data matrix. Find how many PCs are needed to explain 50% of the total variance of the combined data set. Use the results to discuss relations between X and Y. (**C**) Hand-calculate element $c_{3,2}$ of the covariance matrix, what does it represent? Calculate and interpret the full covariance matrix $\{c_{ij}\} = \mathsf{C} = \mathsf{X}\mathsf{Y}^T/N$. (**D**) Perform MCA, how many singular values are needed to explain the covariance between the data sets (this is necessarily a bit subjective)? Explain the significance of the U vectors and V vectors in the MCA analysis. (**E**) Show numerically for these data sets that the *total covariance* is equal to the sum over the singular values squared ($\sum_{i,j} c_{ij}^2 = \sum_j \sigma_j^2$). Conclude from this that σ_j^2 represents the contribution of SVD mode j to the total covariance. Find how many SVD modes are needed to explain 50% of the total covariance of the two data sets. Explain the difference between total variance and total covariance.

* [What's the point of ***optional extra credit challenge problems: apart from the fun of doing them, they may bring the total score of this HW assignment up to 110%, making up for problems you may have missed in this or other HW assignments...]

Python preliminaries & notes

1 python commands within the HW assume you have first used the followings: `import numpy as np; from numpy import linalg; import scipy as scipy; from scipy import linalg; import matplotlib.pyplot as plt; import matplotlib;`
Input a matrix A, column vector **b** and row vector **c** into python in the form
`A=np.array([[a11,a12,a13],[a21,a22,a23],[a31,a32,a33]]; b=np.array([[b1],[b2],[b3]]); c=np.array([[c1,c2,c3]]);` or convert Matlab arrays given in HW directly to python arrays using, e.g., `A=np.array(np.matrixlib.defmatrix.matrix('[1 2 3; 4 5 6]'));`

2 **Hand calculations**, in which you are asked to use only a simple hand-held calculator, are required only when we want to make sure you understand exactly what each step of an algorithm is doing. These also prepare you for the quizzes that involve similar hand calculations. **How much work to show?** Just don't use scratch paper, show all steps that you actually use for the hand calculations, but no more, carrying out calculations to **three significant digits**. Trust the graders to be reasonable, they were students in the course last year. **Examples: (i)** If you are asked, *not* in orange, to calculate the LU decomposition of a matrix, you may use Matlab/python as in `A(2,:)=A(2,:)-A(1,:)*(A(2,1)/A(1,1))` etc, but you may not use a library routine as in `[L,U,P]=lu(A)` except for checking your results. **(ii)** If required to calculate by hand the element $C_{2,3}$ of a matrix product C = AB, you need to explicitly multiply using a hand calculator the second row of A with the third column of B. You may not use Matlab/python to calculate `C=A*B` and then take the needed element from that product, nor to calculate `C23=A(2,:)*B(:,3)`.