CSCE 478/878 Recitation 1 Handout

January 15, 2019

Introduction

For this recitation we will be reviewing linear algebra through the use of NumPy and matplotlib. As with last week, we will be working in and submitting Jupyter notebooks.

A couple general instructions before we start:

- Your Jupyter notebook should be submitted via handin by 4:45PM, Tuesday January 15th.
    - Use the same naming convention as last week: \`<lastname>_<firstname>_2.ipynb\`
- Use markdown cells with **<u>bold underlined</u>** and a font size of 6 titles to denote the start of each problem, e.g.

  <u>Problem 1: Distances and Norms</u>

    - Remember, I will be grading the notebooks directly, so make sure the output you wish for me to see is the last thing ran
- Any question asking for geometric intuition just needs an informal understanding to be shown. No proofs or precise mathematical language are expected.
- You'll probably want to pull up the NumPy Reference manual. In particular, the sections on linear algebra and random sampling will be helpful.

Jupyter Notebook Pointers

Here are a couple tips to (maybe) make your life easier:

- You can run the currently selected cell with \`ctrl + enter\`
    - \`shift + enter\` will both run the currently cell and make a new one
- Tab completion is available in many circumstances
- If you want to clear memory and start over, use the 'Kernel' dropdown menu at the top and select 'Restart'
    - 'Restart and Run All' is a great way to make sure your code works when run from scratch
- You can change the cell type between code and markdown using the 'Cell' dropdown menu
- This article contains several other useful shortcuts and tips
- This article contains info on some of the most common "magic commands" In Jupyter notebooks
    - %system pwd is a nice way of reminding yourself where your notebook is running
    - %matplotlib inline is necessary for displaying your matplotlib pictures in your notebook

---

Problem 1: Distances and Norms

1) In a cell by itself, define a function that accepts two equal sized NumPy arrays (vectors) and returns their cosine similarity. Use NumPy functions to calculate the dot product of the vectors and their magnitudes.

2) In a markdown cell:

a) Write a brief description of what cosine similarity measures focusing on the geometric interpretation
b) Explain the geometric interpretation of two vectors with cosine similarity of 0
c) Explain the geometric interpretation of two vectors with a cosine similarity of 1 or -1

3) Manually define 3 numpy arrays (x1,x2,x3) and randomly generate 1 (x4) such that:
a) Each has 2 dimensions, and is integer valued from the set [-10,10]
b) x1 and x2 are collinear
c) x1 and x3 are orthogonal
d) x4 is randomly generated using NumPy's random module, according to the criteria in a)

4) Print the result of your cosine similarity function applied to the pairs (x1,x2), (x1,x3), and (x1,x4) in a human-understandable way, e.g. "The cosine similarity of [1 1] and [-10 6] is -0.242535"

Note: Cosine Similarity is useful in machine learning when we want a similarity metric between two vectors that is not dependent on their magnitudes. For this reason, it is frequently used in natural language processing to compare the content similarity of documents, even if they differ greatly in length.

5) Use matplotlib.pyplot's "quiver" function to graph all of your vectors on the same plane, assigning each a different color
a) This stackoverflow page has an example you can use as reference
b) You can use np.stack to arrange your data in a similar format to the example

6) Compute and print out the following values for x4: L1 norm (aka Manhattan), L2 norm (aka Euclidean), L4, and L10 norms

7) Compute and print out the following values for the pair of vectors (x1,x4): L1 distance (Manhattan), L2 distance (Euclidean), L4 distance, and L10 distance

Problem 2: Rank, Matrix Inverses, and Solving Linear Systems

1. Using the seed 402, generate a uniform random 3x3 matrix with continuous values from the set [-10,10). We will call this matrix Mat for the remainder of the problem.
   a. Print the matrix
   b. Print the rank of the matrix

2. In a markdown cell, explain the relationship between the rank of a matrix and the existence of the matrix inverse. Explain the geometric intuition for this relationship.

3. Calculate and print the multiplicative inverse of Mat using NumPy.

4. Use the inverse of Mat to solve the equation below for x. Note that * will denote standard matrix multiplication:

$$Mat * x = b$$
$$where\ b = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

   a. Print x
   b. Print the result of Mat * x to prove it equals *b* (Note: using * in NumPy will do component wise multiplication, which is not what we want)

5. Use NumPy's built-in solver (linalg.solve) to find x numerically
    a. Print the new x
    b. Always do this in the real world instead of calculating the inverse first in order to preserve numerical stability as much as possible. It can be an issue on larger problems.

6. Define a new matrix, newMat: $\begin{bmatrix} 1 & -1 & 1 \\ -3 & 2 & 4 \\ -1 & 0 & 6 \end{bmatrix}$
    a. Print newMat's rank
    b. Print newMat's inverse
    c. Print newMat*newMatInverse
    d. Solve the equation in part 4, substituting newMat for Mat
    e. Notice the lack of errors and think about the implications when you may need to use these functions in designing a machine learning algorithm. Write these thoughts in a separate markdown cell.

7. Now approximate the solution for *x* in equation 4 with newMat substituted using linalg.lstsq, which performs least squares approximation
    a. Print the solution
    b. Print newMat*solution, which we will call $\hat{b}$ since it is an approximation of b
    c. Print the Euclidean distance between b and $\hat{b}$

Problem 3: Determinants, Eigenvalues, and Positive Definite Matrices

1. Using a normal distribution and the seed 88, generate a random 3x3 matrix with a mean of 5 and variance of 10. This matrix will be referred to as Mat2 for the rest of the problem.
    a. Print Mat2

2. Calculate the determinant of Mat2, and print it

3. In a markdown box, give the brief geometric intuition for what a determinant means and what it means for a determinant to be negative. What would a 0 determinant mean then?

4. Calculate the eigenvalues and eigenvectors of Mat2, and print them (printing the pair of returned arrays is fine)

5. In a markdown cell, give a geometric understanding of what an eigenvector and its associated eigenvalue represent. What does a 0 eigenvalue mean?

Grading

Parts 1 and 2 will be worth 40% each. Part 3 will be worth 20%. You will be graded on the output, markdown text, and code in the Jupyter notebook as it is when it is handed in.