PROBLEM SET 1: APPLIED MATHEMATICS 216

Due: Friday February 4 at 11:59pm

Goals for the week.

(1) Get some experience with coding in python. For those of you who do not know python, this is likely to be the main challenge in carrying out the homework problems below. The class notebook contains lots of background on how the functions work. Play with them! Start by looking for commands (plotting, etc.) on Stack Overflow. Don't be nervous to ask your classmates and us for help. Remember, instead of teaching 'coding from scratch', our aim is to get you all up to speed with the software tools we are using by example.
(2) Understand qualitatively how different types of norms can influence the results of regression problems, especially the robustness to outliers. Estimation of the errors in computing the coefficients.
(3) Get a baseline understanding of PCA and SVD
(4) For linear regression, $Ax = b$, where A is a $n \times m$ matrix, x is a column vector of size m and b is a column vector of size n, count the number of arithmatic operations that are required to solve the problem, assuming the problem is overdetermined $m > n$.
(5) Learn about convolving and deconvolving images. An essential part of experimental science is image collection and analysis. Once obtained images often require different forms of manipulation to be useful.

Problems.

(1) *What polynomial did we pick?*

We have posted on the course web site one datafiles, called "Problem1.txt" containing 1000 points (x, y) that we have drawn from a tenth order polynomial of the form:

$$y = \sum_{i=1}^{10} a_i x^i$$

We have also added noise to the data that was chosen from one of the distributions that is a method in numpy.random (see here) for a list.

Your challenge is to solve the inverse problem and figure out which polynomial we have chosen. Your answer should give the coefficients a_i, and also your estimated errors in the coefficients ($a_i \pm \epsilon_i$).

(2) *Beyond polynomials*

The problem above was easy in that we actually told you which functions we expanded in. Life is not usually so easy. For this problem, we are providing a datafile "Problem2.txt" that 1000 samples of the form (x, y) from the function $y = f(x)$.

Your goal here is to determine, to the best of your ability, what the function $f(x)$ is. Once complete, please submit your predictions on the test set to Kaggle. The Kaggle link can be found here. There will be more Kaggle competitions during the semester and the best performers will win prizes.

(3) *Linear Regression and Counting operations*

In class we discussed the linear regression problem

$$Ax = b,$$

where A is an $m \times n$ matrix, x is a column vector of length n, and b is a column vector of length m. Let's assume that $m > n$ so the problem is overdetermined. Recall what m and n represent. We construct the matrix A by transposing each of our points from our data set and stacking them on top of each other. m therefore represents the number of data points in our training set and n is the dimensionality of our input space.

As discussed in class, in this case we can find x by minimizing the loss function

$$\mathcal{L}[x] = ||Ax - b||^2.$$

(a) Derive the equations asserted in class, that the solution obeys

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

Note that the derivation is also given in the video linked on the Canvas page.

(b) Estimate the number of floating point operations to compute the following quantities. We are particularly interested in how they depend on m and n
   (i) $A^T A$
   (ii) $A^T b$
   (iii) $(A^T A)^{-1}$.
   (iv) Total floating point operations to compute x.

An alternative way of solving this optimization problem is to use **gradient descent**. The idea is to start out with a guess for the solution x. Let's call this guess x_0. We then can calculate a new guess by demanding that the value of the loss function decreases. We can guarantee this by moving down the gradient, namely by demanding

$$x_1 = x_0 - \gamma \nabla \mathcal{L}.$$

We can then iterate

$$x_n = x_{n-1} - \gamma \nabla \mathcal{L},$$

until $\mathcal{L}$ decreases enough for us to stop.

(a) Show that $\nabla \mathcal{L} = A^T Ax - A^T b$.
(b) Compute the number of operations for doing a single step of gradient descent.
(c) Suppose that it takes 10 iterations for gradient descent to converge to an acceptable answer. Compute the total number of operations for doing this many iterations. (Note: It helps to precompute the matrix $A^T A$ so that you only have to do this once. Then use this for the subsequent iterations.)
(d) Now compare the total cost of solving the problem with gradient descent relative to the cost of using the direct method. Which is better?
(e) **Extra Credit** Ten iterations might not be enough. In class on Monday January 31, we will go through some code implementing the gradient descent method. Use this code to construct an argument decide how many iterations are enough for accurate convergence. Reassess your answer to this question in this limit.
(f) **Extra extra credit** Another variant of gradient descent is *stochastic gradient descent*. This is especially useful when we have a large amount of data ($m \to \infty$) so that even computing the gradient is very expensive (because of the cost of $A^T A$). We will discuss stochastic gradient descent later on in class – but get ahead and figure out whether as

$m \to \infty$, whether stochastic gradient descent is a more efficient way of solving linear regression than any of the methods described here.

(4) *Images!* Systems where a single input maps onto a single output are nice, but in science it is often convenient to use more complex representations for examining data. One such example are images. Commonly, when given an image, there is a need to refine and manipulate it. In this problem we provided a tutorial that explains a particulary powerful manipulation method: convolution. A common definition of the two-dimensional discrete convolution C of an image array I with a filter array F is:

$$C_{i,j} = \sum_{k,l} F_{k,l} I_{i-k,j-l}$$

In this problem, you will play with various manipulations of images.go through the steps of compressing an image through both principle component analysis and singular value decomposition. You will analyze the impacts on memory saving these methods have, and then play with methods for deconvolving images in skimage. Finally you will try your hand at deconvolving an image that we have distorted. The details of this problem are contained int he notebook for Problem 4.

You will be given an image of Tom Holland that has been convolved and noised in an unknown manner. You'll need to restore it at the best of your abilities.

Submission Instructions.

(1) Submit 2 notebooks to Canvas:
LASTNAME_FIRSTNAME_P1_2
LASTNAME_FIRSTNAME_P3
(2) Submit your predictions to Kaggle