Homework 4

Please scan and upload your assignments on or before March 23, 2018.

- You are encouraged to discuss ideas and collaborate with each other; but
- you must *clearly* acknowledge your collaborator, and
- you must compose your own writeup and/or code independently, and
- you must combine all derivations, code, and results as a single PDF to be uploaded on Canvas.

---

1. **(15 points)** In class, we analyzed the per-iteration complexity of k-means. Here, we will prove that the k-means algorithm will terminate in a finite number of iterations. Consider a data set $X = \{x_1, \dots, x_n\} \in \mathbb{R}^d$.

   a. Show that the k-means loss function can be re-written in the form:

   $$F(\eta, \mu) = \sum_{i=1}^{n} \sum_{j=1}^{k} \eta_{ij} \|x_i - \mu_j\|^2$$

   where $\eta = (\eta_{ij})$ is a suitable binary matrix (with 0/1 entries). Provide a precise interpretation of η.

   b. Show that each iteration of Lloyd's algorithm can only decrease the value of F.

   c. Conclude that the algorithm will terminate in no more than T iterations, where T is some finite number. Give an upper bound on T in terms of the number of points n.

2. **(10 points)** PCA is only one example of expressing a data matrix into a product of factors; see notes for Lecture 15. There are many other such decompositions; for example, nonnegative matrix factorization (NMF) solves the following optimization problem:

   $$\min_{H \in \mathbb{R}^{r \times d}, W \in \mathbb{R}^{n \times r}} f(H, W) := \frac{1}{2} \|X - WH\|_F^2,$$
   $$s.t. \quad H \geq 0 \quad and \quad W \geq 0,$$

   where $H \geq 0$ (and $W \geq 0$) requires all elements of H (and W) to be *nonnegative*, and $\| \cdot \|_F$ is the Frobenius norm. There is a different algorithm, known as Multiplicative Updates (MU), which solves the NMF problem as follows:

Input: Data X.

Output: $H^* \geq 0, W^* \geq 0$, s.t. $X \approx W^* H^*$.

(Step 1): Initialize $H_0 > 0, W_0 > 0, t = 0$.

(Step 2): Repeat until convergence:

i. $H_{ij}^{t+1} \leftarrow H_{ij}^{t} \frac{((W^t)^T X)_{ij}}{((W^t)^T W^t H^t)_{ij}} \quad i = \{1, \dots r\}, \quad j = \{1, \dots d\}$.

ii. $W_{ij}^{t+1} \leftarrow W_{ij}^t \frac{(X(H^t)^T)_{ij}}{(W^tH^t(H^t)^T)_{ij}} \quad i = \{1, \dots n\}, \quad j = \{1, \dots r\}.$

iii. Increment t.

- **Note:** The initial estimates H_0, W_0 are carefully chosen such that all elements are nonnegative.

   a. Evaluate the gradient of the loss function $f(H, W)$ with respect to H, i.e. $\nabla_H f(H, W)$.

   b. Evaluate $\nabla_W f(H, W)$ (using the symmetry of the problem and your answer to part 2.a., or explicitly).

   c. Show that the update steps in (Step 2).i. (and (Step 2).ii.) can be expressed as:

$$\begin{aligned} H_{ij}^{t+1} &= H_{ij}^t - \alpha_H^t \nabla_H f(H^t, W^t), \\ (and) \quad W_{ij}^{t+1} &= W_{ij}^t - \alpha_W^t \nabla_W f(H^t, W^t), \end{aligned}$$

   where $\alpha_H^t = g_1(H^t, W^t)$ and $\alpha_W^t = g_2(H^t, W^t)$, are functions of H^t, W^t. What are the step size functions g_1 and g_2 ?

   d. Comment on whether the updates H^{t+1} and W^{t+1} satisfy the non-negativity constraint.

3. **(10 points)** In this experiment, we will use the singular value decomposition as a tool for compressing raw image data. This is not how images are actually compressed; for example, JPEG compression algorithms do more fancy (and interesting) computations. However, the idea is similar: if we are willing to tolerate a certain amount of distortion, then we can get away with a much more concise data representation.

   a. Read the "Mandrill" image file (provided as a supplement to this assignment), and convert it into grayscale by averaging the R,G,B values for each pixel. Your image is now a 298×298 matrix; call it X.

   b. Perform an SVD of X to obtain the decomposition $\{U, \Sigma, V\}$. Plot the singular values (i.e., the diagonal entries of Σ) in decreasing order.

   c. Choose $k = 10$, and reconstruct an approximation of X using the top k singular values and vectors, U_k, V_k, and Σ_k. Display this approximation, and calculate how many numbers you needed to store this approximate image representation. Divide by the original size of X to get the *compression ratio*.

   d. Repeat this experiment for $k = 20, 30, 40, 50, 60$. Display these images, and report their compression ratios in the form of a table. Is there any benefit in going for higher k?

4. **(15 points)** Standard raw images are stored using an RGB encoding with 1 byte per color per pixel (i.e., one number between 0 and 255 for each color channel). Therefore, every pixel can be interpreted as a point in a *color space* with $d = 3$ dimensions, and any given image with n pixels can be viewed as a data set with n samples. Richly colored images represent data sets that are very dispersed in the color space. The process of *color quantization* is to simplify the palette of the image. Using a parameter k, we can replace each RGB pixel with one out of k "representative" colors that still preserve the overall image quality. In other words, we restrict pixels to a discrete set of k colors.

   a. Consider the "Palace" image given in this assignment. Fix $k = 8$, and choose the *cubic* quantization scheme where the representative colors are chosen from the set $\{(0, 0, 0), (0, 0, 255), (0, 255, 0), \dots, (255, 255, 0), (255, 255, 255)\}$. Replace each pixel with its nearest (Euclidean) neighbor among these points to get a new image Display both the original image and the new version side by side.

b. Now, think of the image as a dataset with n samples, and run k-means on this dataset to get k cluster centers. Most existing scientific programming environments (Matlab, R, Python, etc) will have a good implementation of k-means; but coding it up yourself shouldn't be too hard either. Replace each pixel with its cluster center, and display both the image and its quantized version side by side. This procedure can be viewed as a data-dependent color-quantization scheme.

c. Repeat the experiment for $k = 27$ and $k = 64$. Once again, compare uniform quantization versus k-means. For example, with uniform quantization and $k = 64$, choose the quantization levels in each channel as $[0, 85, 170, 255]$.

5. **(Optional)**: how much time did you spend on this assignment?