Homework #1: Eigenflowers (Feature Spaces and Principal Component Analysis)

Solutions to this assignment are to be submitted in Brightspace via Assignments. The submission deadline is **Saturday September 27, 2025 at 11:59pm**. You should submit am all inclusive Jupyter notebook with any code you write and any explanations as Markdown comments. All code, plots and descriptions should be in the notebook.
PLS DO NOT USE GOOGLE COLAB for this assignment.

This assignment walks through an end-to-end PCA pipeline on a flowers image dataset (obtained from the Oxford 102 Flowers dataset online). We will be using only a small portion containing 6 classes with 30 training images. Below is an overall summary of what you will do for this assignment:

- Preprocess images (grayscale, resize, flatten)
- Build the data matrix ‘A‘ (pixels $\times$ samples)
- Compute PCA via **thin SVD**
- Then do the same and via the **dual trick** ($L = \frac{1}{M} A^\top A$) using the in-build function `eigen` in texttnumpy.linalg.
- Visualize the **mean image** and **top eigenimages(or eigenflowers)**
- Review the EVR
- Reconstruct images (from the train set and the never-been-seen-before test set) using the top-k components and evaluate the error

1. The first two sections contains your imports and setting up your data directory and other inputs. We will use the standard scientific Python packages (`numpy, PIL, matplotlib`) along with `hashlib`. Note that each student should have their own unique seed for repeatability. Make sure your kernel uses Python ≥ 3.8.
2. **Preprocessing helpers:** We have small utilities for you to: Walk the dataset directory; Convert your colored images to grayscale; Resize all images to the same size; Flatten to vectors and normalize to [0,1]; and Stack as columns into the data matrix A of shape (pixels, M)
3. **Load and preprocess images:** This should be pretty fast because we are only loading 30 flower images. Print out the shapes of your images to ensure you have loaded them correctly.
4. (5 points) **Mean image and centering:** Compute the mean image across columns of A; Subtract it from each column to get A_{centered}. An example of what the mean image looks lie is shown in Figure 1

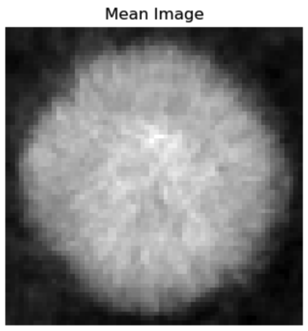

Figure 1: The mean image reshaped and displayed

5. (10 points) **Implement PCA via Thin SVD (Method A):** Compute the SVD of A_{centered} directly: You can use the SVD function `eigh` from the `linalg` library in `numpy`.

6. (5 points) **Implement PCA via the dual trick:** Compute eigenpairs of the small sample–sample L matrix given A_{centered}. Similar to the previous question, solve directly for the eigenvalues and eigenvectors (known as the eigenpairs) of L using the prebuilt prebuilt linear algebra `eigvals, V = np.linalg.eigh` function call.

7. (20 points) **Dual trick + eigendecomposition using the power method:** Again we want to compute eigenpairs of the matrix, but this time we will use our own implementation of the power method. The pseudocode for doing this is given in the `.ipynb` file. The process is split into two functions where the first function `power_method_sym` returns an approximate value the dominant eigenvalue and the corresponding eigenvector of a matrix L. The second function `power_method_sym_allk` returns the first k eigenvalues and eigenvectors.

8. (5 points) **Compute your eigenimages/eigenflowers** using each of the 3 PCA methods (1:thin SVD, 2:dual method, 3:dual with power method). Make sure all 3 methods have the same shape. If done correctly, all three methods show have very similar results.

9. (5 points) **Display the top k eigenflowers** Below are the top 6 top eigenflowers (the fist 6 principal components) in my implementation.

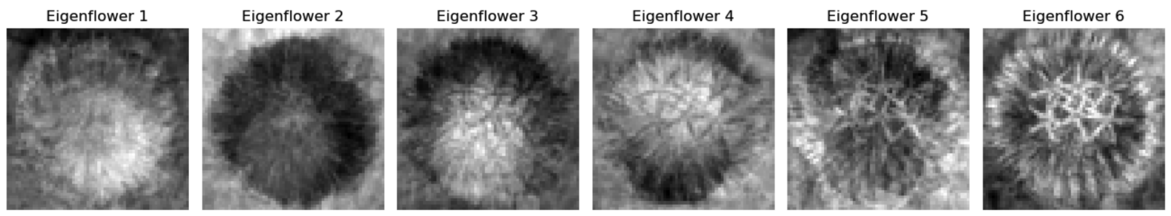

Figure 2: The first 6 principal components or eigenflowers, reshaped and displayed.

10. (10 points) **Reconstruction with top-k components** Pick a flower image from the training set and project it onto only the top k eigenimages, and reconstruct. You pick your k. Display your reconstructions (code to do this is provided). Observe how reconstruction improves as k increases. Below is my reconstruction with changing values of k.

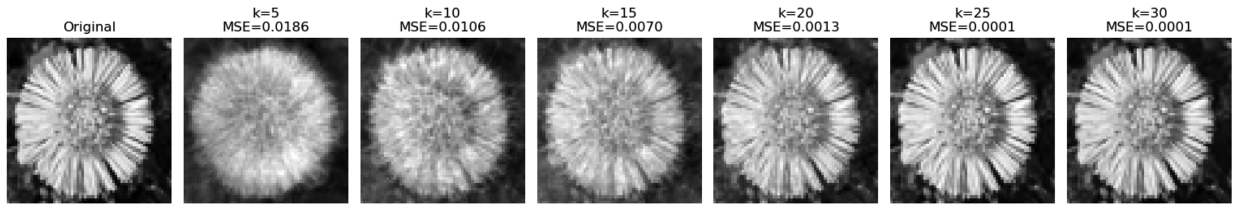

Figure 3: Left: original; next six images are the reconstructions with increasing k values.

11. (10 points) **Explained variance ration (EVR):** is a statistic used in Principal Component Analysis (PCA) (and other dimensionality reduction techniques) to show how much information (variance) each principal component captures from the original dataset. The formulas to compute the EVR and the cumulative EVR is given in the notebook. Follow the steps and compute and plot the cummulative EVR. Below is an example of my cummulative EVR.

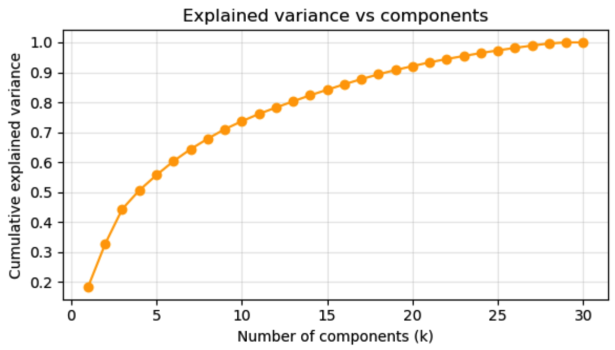

Figure 4: Cummulative EVR vs. components

12. (10 points) **Feature mapping:** Explain how this approach can be used for feature extraction (do this in the comments section of the notebook). What exactly are the features and how do we obtain them here?

13. (20 points) **Getting familiar with common distributions from the exponential family** (you can submit this in a separate file and zip your submission)
Write out the mathematical formula for each of the distributions below, stating which variable(s) represents the parameter(s) of the distribution. Indicate whether it models discrete or continuous variables. Also, in one or two sentences, describe when this distribution would come in handy during a modeling task. They are:

(a) Gaussian; (b) Bernoulli; (c) Binomial; (d) Multinomial; (e) Exponential; (f) Poisson.

Bonus (5 points): Plot the T-SNE map of all the training data vectors in A; you can add this at the end of your submission notebook.