Denoising with SVD

Applied Linear Algebra

Course Instructor: Peyman Adibi
Alireza Dastmalchi Saei
(993613026)

Introduction

In this project the aim is to explore the effectiveness of Singular Value Decomposition (SVD) for denoising images. We will add noise to an image dataset from Kaggle (architecture) and then apply SVD to denoise it with different number of singular values (K).

Code Explanation

First, we import the necessary libraries for this project:

Required libraries

```
from PIL import Image
import os
import numpy as np
import random
import math
import matplotlib.pyplot as plt
import cv2
```

Then we use a function to read images from the architecture folder of the dataset in Kaggle:

Function to read images from given path

+ Code + Markdown

```
def read_images(directory):
    image_data = []
    for file in os.listdir(directory):
        if file.endswith(".jpg"):
            image_path = os.path.join(directory, file)
            image = Image.open(image_path)
            pixel_data = np.array(image)
            image_data.append(pixel_data)
    return image_data

# 128 * 128 images
directory_path = "/kaggle/input/image-classification/images/images/architecure"
image_data_list = read_images(directory_path)
print(f'Number of images: {len(image_data_list)}')
```

```
Number of images: 8763
```

In the next part, a gaussian noise must be added to the picture before denoising it:

```
def gaussian_noise(shape, mean=10, std_dev=10):
    size = np.prod(shape)

    u1 = np.random.uniform(0, 1, size)
    u2 = np.random.uniform(0, 1, size)

    # Apply the Box-Muller transform to generate Gaussian noise
    z = np.sqrt(-2.0 * np.log(u1)) * np.cos(2.0 * np.pi * u2)

    # Reshape the noise array to the desired shape
    z = np.reshape(z, shape)

    # Use standard deviation and mean
    noise = std_dev * z + mean

    return noise.astype(np.uint8)
```

In the next step, the noise is added to the RGB values of the pictures and it is also checked that the RGB values must in range [0, 255] making the image noisy.