Tutorial: Image Compression using Singular Value Decomposition

Song Liu (song.liu@bristol.ac.uk)

GA 18, Fry Building,

Microsoft Teams (search "song liu").

Image Compression

- One usage of Singular Value Decomposition (SVD) is compressing large matrices.
- We have learned that images are essentially matrices of numeric values when stored in computers.
- Therefore, we can use SVD to compress images.

SVD

Recall, SVD of a matrix $M \in \mathbb{R}^{m \times n}$ finds the following three matrices:

- $U \in \mathbb{R}^{m \times r}$,
- $D \in \mathbb{R}^{r \times r}$ is a diagonal matrix stores singular values,
- $V \in \mathbb{R}^{n \times r}$,

  such that

$$M = UDV^\top$$

and $r = \min(m, n)$.

In numerical softwares (such as R or MATLAB), the singular values stored in D **are sorted decreasingly**.

SVD Compression

Suppose singular values in D are sorted decreasingly,
SVD can be used to construct an approximation of M:

$$M_1 = U_1 D_1 V_1^\top,$$

where

- $U_1 = U_{[1:m, 1:r_1]}$,
- $D_1 = D_{[1:r_1, 1:r_1]}$,
- $V_1 = V_{[1:n, 1:r_1]}$.
- r_1 is a positive integer smaller than r.

Loading Images

Install "imager" package if you have not.

```
install.packages("imager")
```

Load an image into the matrix `M`.

```
library(imager)
img <- load.image("UoB.jpg")
img <- grayscale(img)
M <- as.matrix(img)
```

Now `M` should contain an matrix whose entries are pixel values of the image.

Checking out the Image

Plot the image

```
plot(as.cimg(M))
```

Check out how much memory does it take to store this image:

```
# fill out the blank.
size1 <- ______
print(paste("size:", size1, "bytes"))
```

Compression

Now, use builtin `svd` function to obtain U, D, V for `M`.

- Hint: `?svd`

Double check you have used `svd` correctly:

```
norm(M - U%*%D%*%t(V), type = "F")
[1] 1.787766e-12
```

Check out r:

```
dim(U)[2]
674
```

Compression

Let us $r_1 = 100$.

Construct U_1, D_1, V_1 using U, D, V and r_1.
Reconstruct the M_1 using U_1, D_1, V_1.

Examine the Compression

Plot the compressed image:

```
plot(as.cimg(M_1))
```

Does it look like Netflix when set to low quality?

Check out how much memory does it take to store the compressed image:

```
# Fill out the blank
size2 <- ____
print(paste("size:", size2, "bytes"))
```

What is the compression ratio `size2/size1`?
Do you think given the image quality degradation, such a compression is worth it?

Advanced Mathematical Question:

Why is M_1 called an approximation of M?

- Hint: take the difference $M - M_1$ and check the reminder.

Our construction M_1 is the best "low rank approximation" of M in terms of Frobenius norm.

- Read: https://en.wikipedia.org/wiki/Low-rank_approximation
- Low rank approximation is a classic problem in machine learning.