

CENG 371  
Scientific Computing  
Homework 4  
BARTU KILIÇKAYA / 2380640

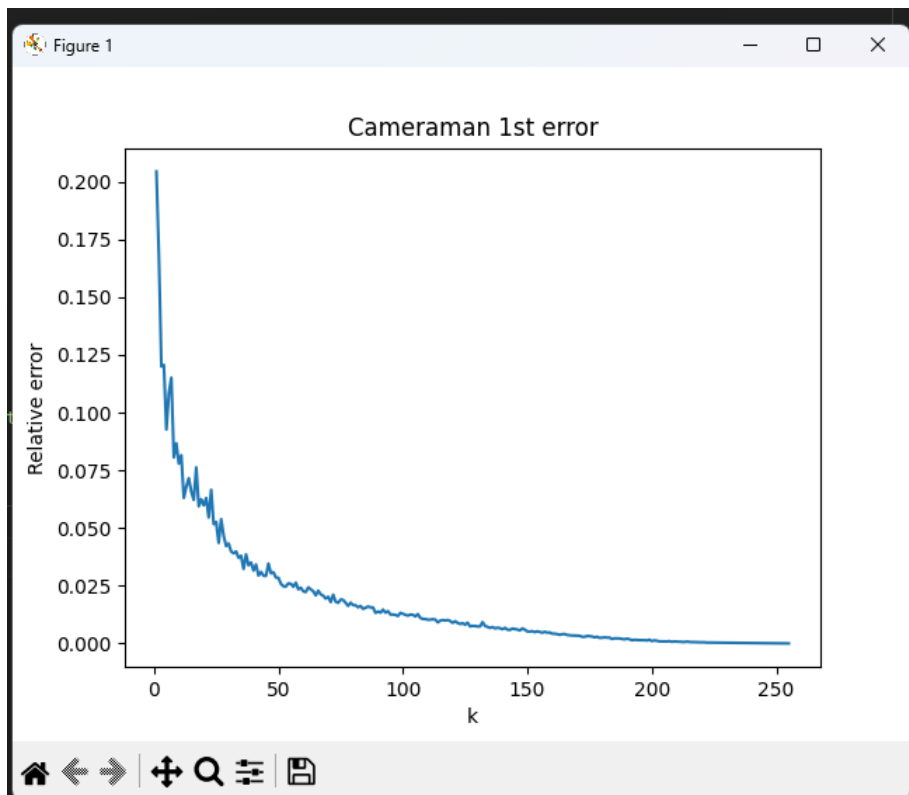
1)

Function is implemented as **approximate\_svd(A,k)** with parameter  $p=5$ , returns  $u_k, \sigma_k, v_k$  respectively.

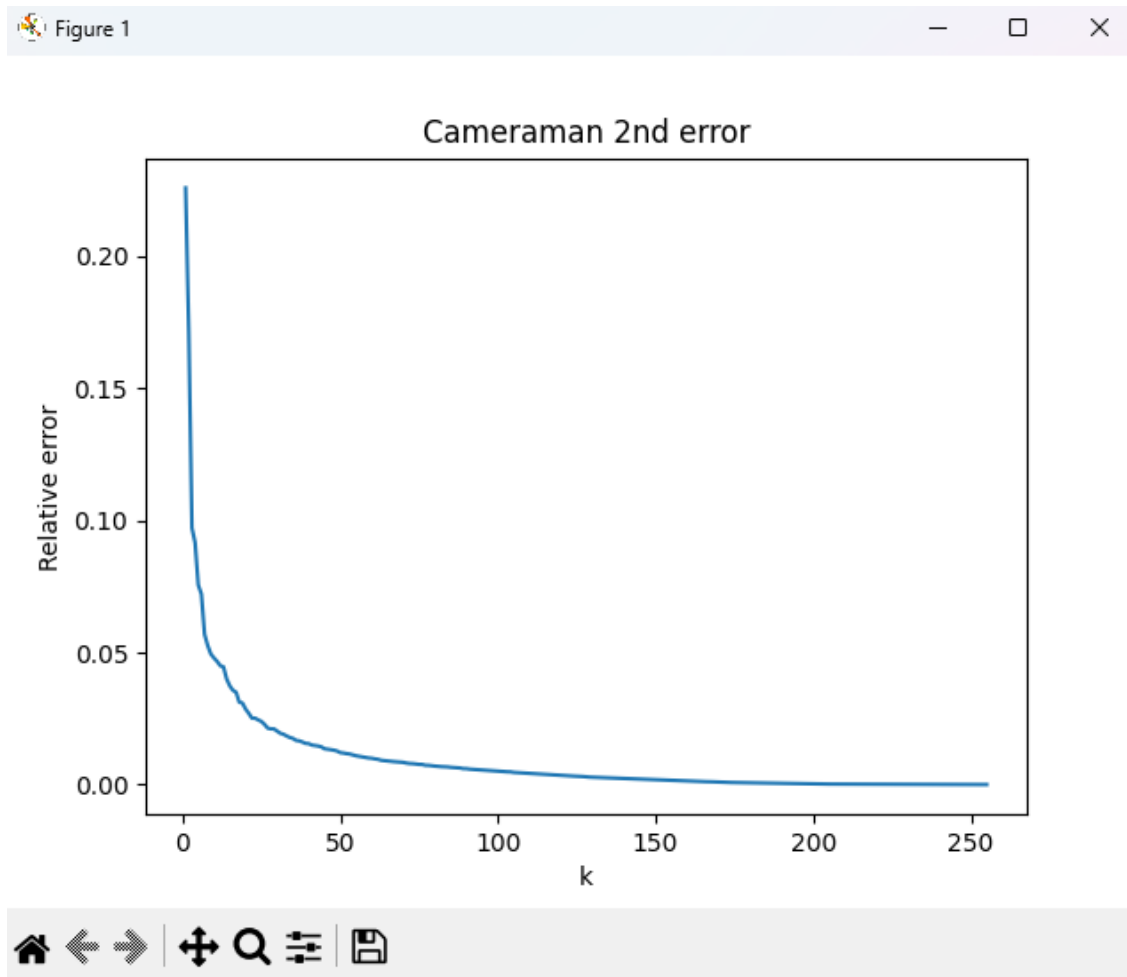
2)

a) Cameraman:

1<sup>st</sup> error:

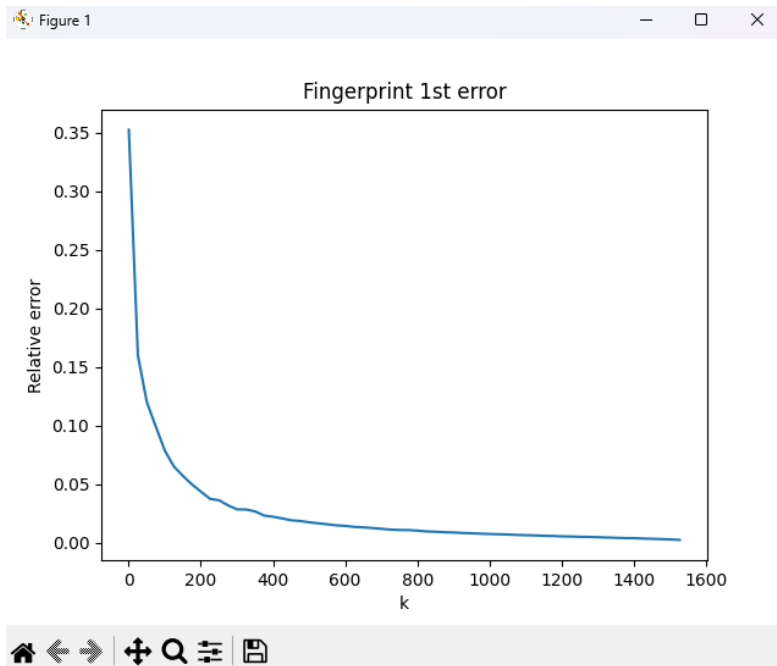


2<sup>nd</sup> error:

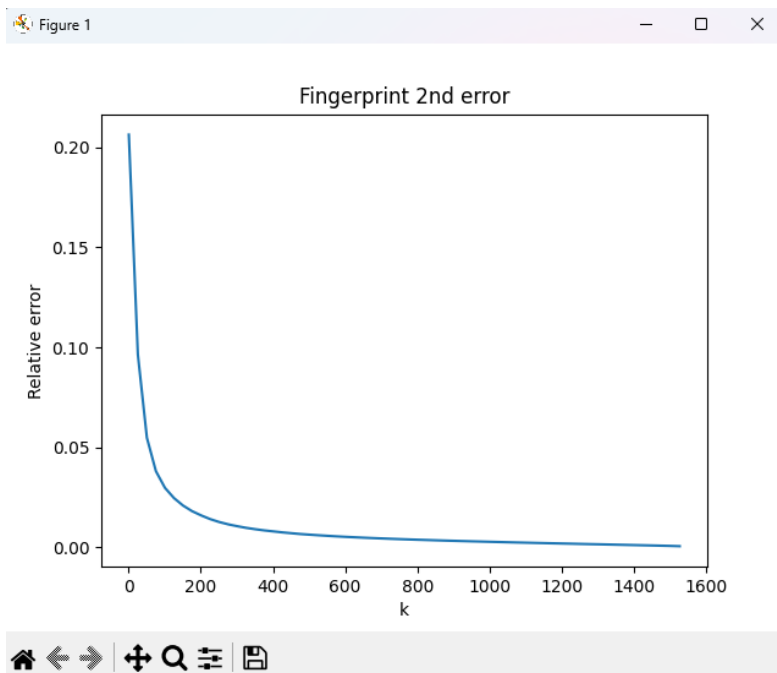


Fingerprint:

1<sup>st</sup> error:



2<sup>nd</sup> error:

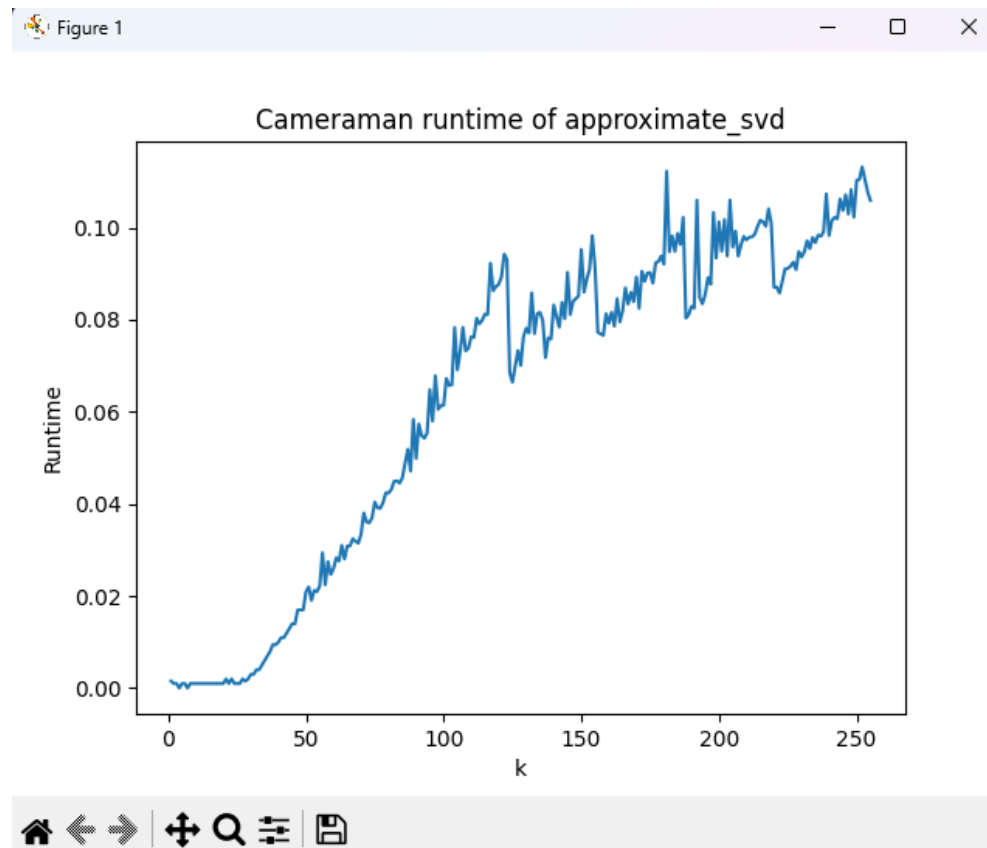


In both cameraman and fingerprint, as  $k$  increases, relative error decreases as expected. Svds relative error curve has more steep decline

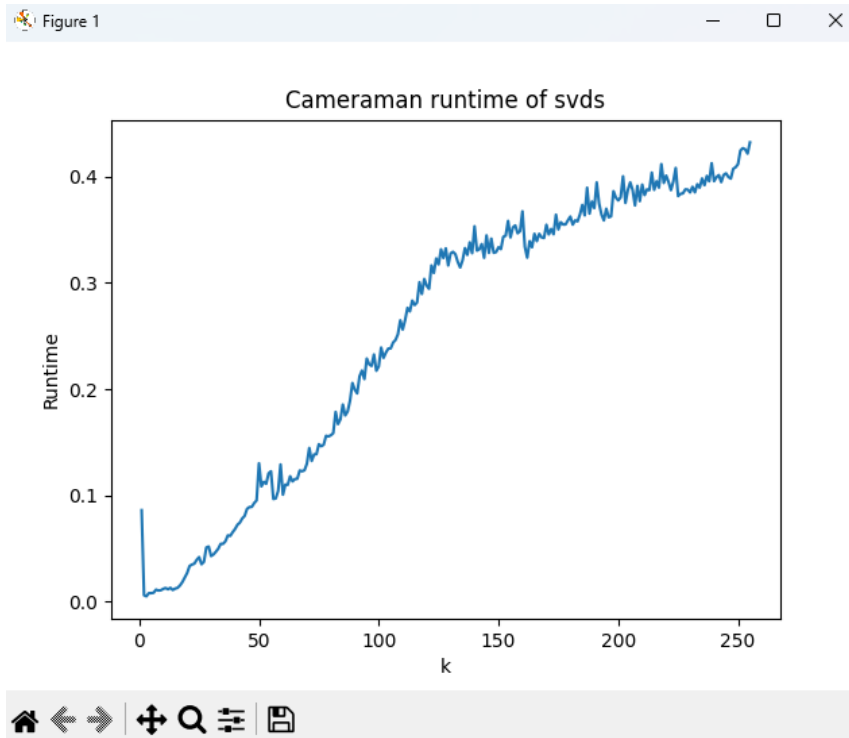
than `approximate_svd`, so lower  $k$  values in svds are closer to 0 than `approximate_svd`.

**b) Cameraman:**

Cameraman runtime of `approximate_svd`:

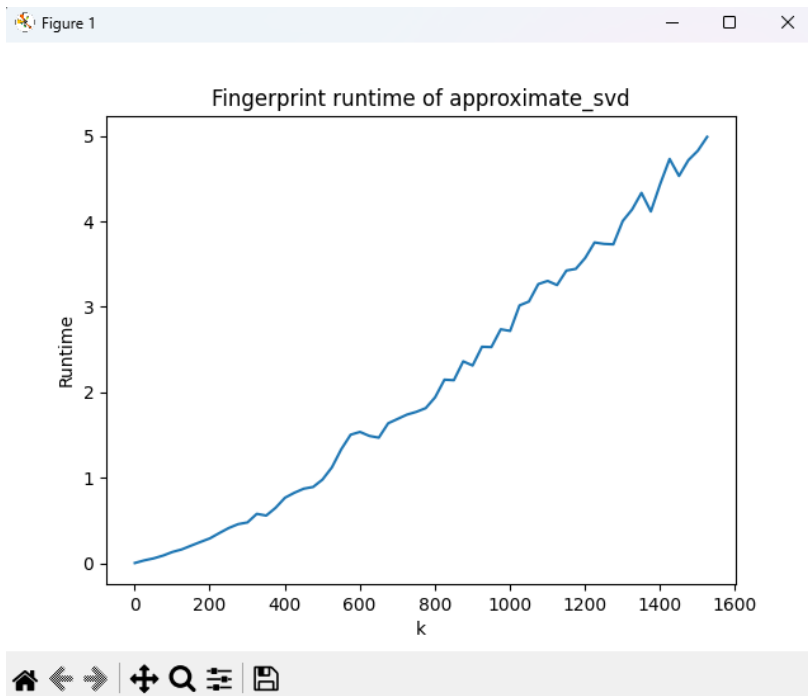


Cameraman runtime of svds:

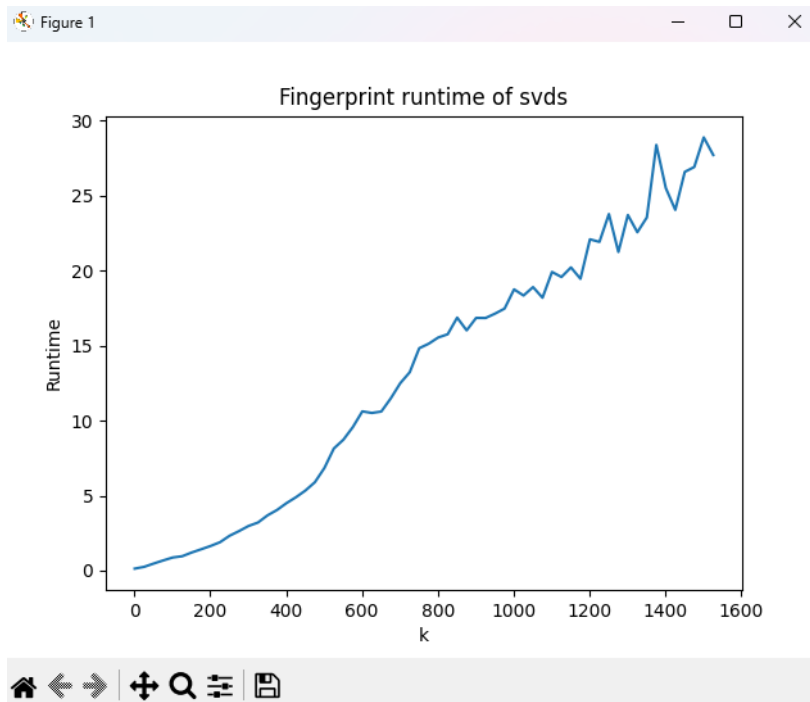


Fingerprint:

Fingerprint runtime of approximate\_svd:



## Fingerprint runtime of svds:



For runtime comparison, fingerprint runtimes are bigger than cameraman runtimes as expected since fingerprint's matrix is way larger than cameraman's. Additionally, `approximate_svd` is faster than `svds` in both images.

c)

Setting  $k=25$  for both images;

Camerman approximate\_svd:



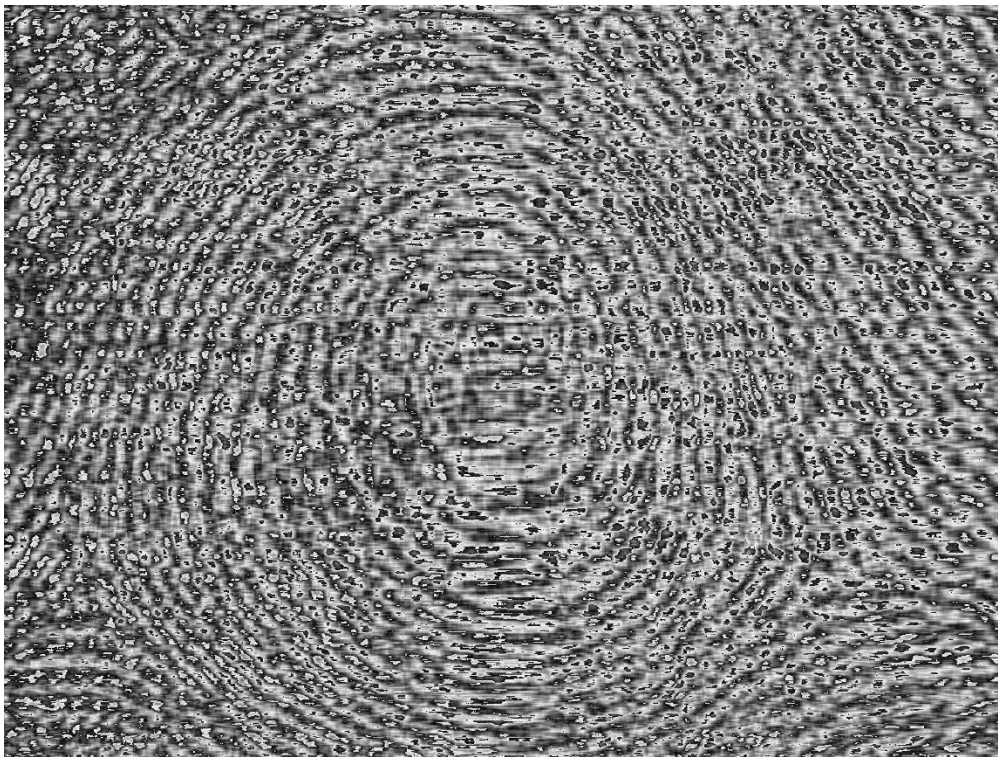
Camerman svd:



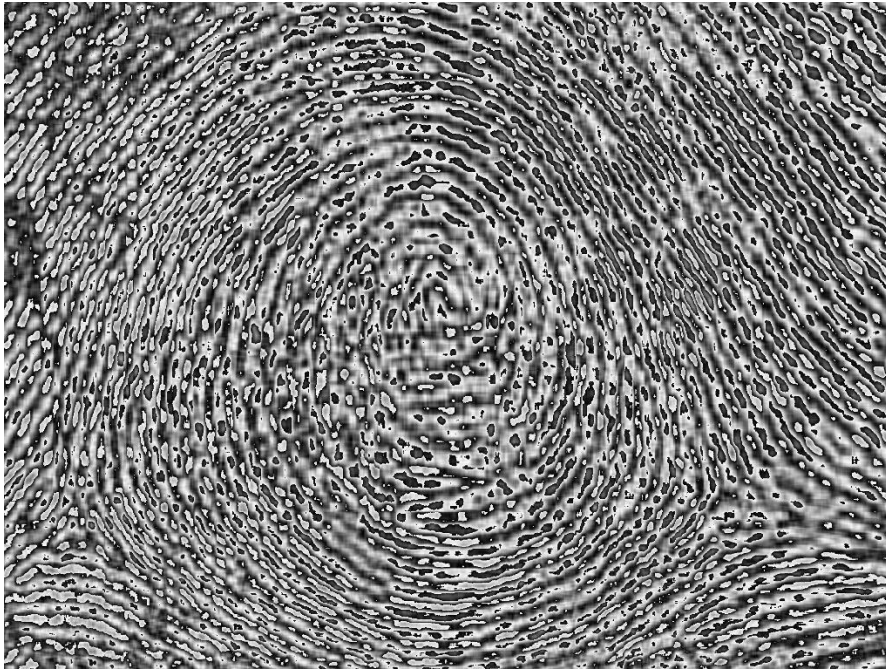
Camerman svds:



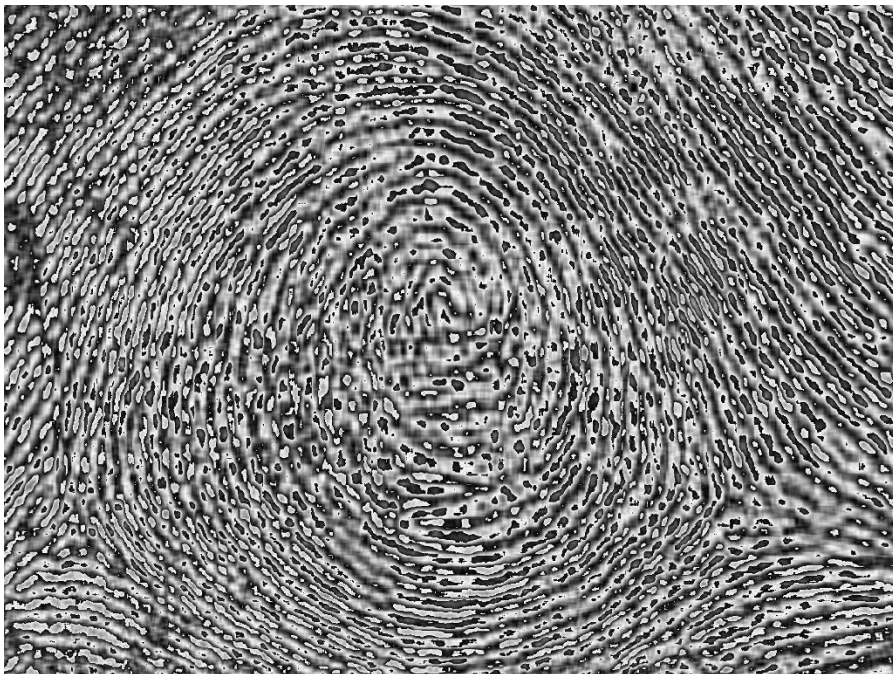
Fingerprint approximate\_svd:



Fingerprint svd:



Fingerprint svds:



In both images, we get the same result in svd and svds. Even recompiled image sizes are the same. However `approximate_svd`

results are different from svd and svds. Svd and svds images are closer to original images than the images recompiled from `approximate_svd`.

IF we select  $k$  close to the rank of the matrix, we get the image almost the same as the original image.

**d)**

Since `approximate_svd` runs faster than both svd and svds, for large image files, it can be used to approximate. Even though svd and svds give better results than `approximate_svd`, we may encounter really large matrices and using `approximate_svd` with appropriate  $k$  value will be better if time is an issue. It is particularly useful when the matrix is too large to fit in memory, or when only a small number of the singular values and vectors are needed. In contrast, the standard svd and svds algorithms compute the full svd of a matrix, which can be computationally expensive for large matrices.