MAT185 – Linear Algebra
Assignment 2

Instructions:

Please read the **MAT185 Assignment Policies & FAQ** document for details on submission policies, collaboration rules and academic integrity, and general instructions.

1. **Submissions are only accepted by** Gradescope. Do not send anything by email. Late submissions are not accepted under any circumstance. Remember you can resubmit anytime before the deadline.
2. **Submit solutions using only this template pdf**. Your submission should be a single pdf with your full written solutions for each question. If your solution is not written using this template pdf (scanned print or digital) then your submission will not be assessed. Organize your work neatly in the space provided. Do not submit rough work.
3. **Show your work and justify your steps** on every question but do not include extraneous information. Put your final answer in the box provided, if necessary. We recommend you write draft solutions on separate pages and afterwards write your polished solutions here on this template.
4. **You must fill out and sign the academic integrity statement below**; otherwise, you will receive zero for this assignment.

Academic Integrity Statement:

Full Name: Jessica (Xuezheng) Fu

Student number: 1011048607

Full Name: Sara Parvaresh Rizi

Student number: 1010913451

I confirm that:

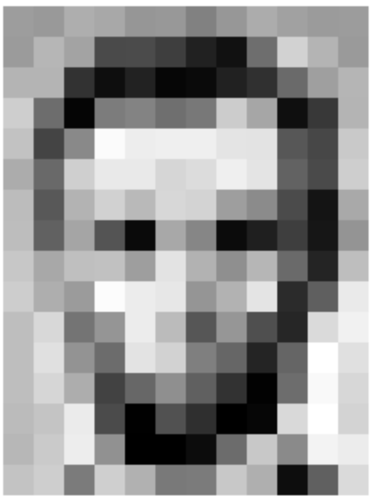
- I have read and followed the policies described in the document **MAT185 Assignment Policies & FAQ**.
- In particular, I have read and understand the rules for collaboration, and permitted resources on assignments as described in subsection II of the the aforementioned document. I have not violated these rules while completing and writing this assignment.
- I have not used generative AI in writing this assignment.
- I understand the consequences of violating the University's academic integrity policies as outlined in the Code of Behaviour on Academic Matters. I have not violated them while completing and writing this assignment.

By submitting this assignment to Gradescope, I agree that the statements above are true.

Preamble:

Image processing is the manipulation of digital images by applying mathematical tools and algorithms. A wide range of applications based on digital image processing are, for example, medical imaging, image optimization in consumer cameras, computer vision, and satellite imagery. Linear algebra, and the techniques you will learn during this course, plays a crucial role in that field by providing a mathematical foundation.

A typical digital image can be considered as a 2-dimensional matrix of *pixels* (abbreviation for *picture element*). Methods of digital image processing are manipulating this 2D-matrix. A pixel is the smallest element of a digitally acquired raster image, and can be considered as a colour sample at each point of an image. Typically, each pixel is represented by 3 positive integer values from 0 to 255 for each colour red, green, and blue: 0 is representing black and 255 ($= 2^8 - 1$) either red, green, or blue. In that case, 24 Bits are used to code 16,777,216 distinct colours for each pixel. This is called a 24 bpp (24 Bits-per-pixel) colour depth. If a grey-scale image is sampled, each pixel samples the light intensity. In that case, only 8 Bits (single integers from 0 to 255) are typically necessary, as shown in Figure 1, to store grey-scale images.



157	153	174	168	150	152	129	151	172	161	156	156
156	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	257	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
206	174	156	252	236	231	149	178	228	43	96	234
150	216	116	149	236	187	86	150	79	38	218	241
150	224	147	108	227	210	127	102	36	101	256	224
190	214	173	66	103	143	56	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	256	211
183	202	237	145	0	0	12	108	250	138	243	236
196	206	123	207	177	121	123	200	175	13	96	218

157	153	174	168	150	152	129	151	172	161	156	156
156	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	228	227	87	71	201
172	106	257	233	233	214	220	239	228	98	74	206
188	88	179	209	185	215	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	168	191	193	158	227	178	143	182	106	36	190
206	174	156	252	236	231	149	178	228	43	96	234
150	216	116	149	236	187	86	150	79	38	218	241
150	224	147	108	227	210	127	102	36	101	256	224
190	214	173	66	103	143	56	50	2	109	249	215
187	196	236	75	1	81	47	0	6	217	256	211
183	202	237	145	0	0	12	108	250	138	243	236
196	206	123	207	177	121	123	200	175	13	96	218

Figure 1: Grey-scale image represented by 8 bpp (Bits-per-pixel). [*source: stanford.edu*]

One application of linear algebra in the context of digital image processing is for a linear transformation of the images / 2D-matrices. We will discuss linear transformations in more detail later this term. Common transformations include scaling, rotation, and translation of the image. All of these linear transformations can be represented by a matrix multiplication.

Another important application of linear algebra in the context of digital image processing is filtering, which will be explored in Question 1 of this assignment. Filtering includes, for example, methods for noise reduction, blurring/sharpening, edge detection, white balancing, colour correction, and many more.

Lastly, images have to be stored in efficient ways. For a colour image of size $n \times m$ with a 24 bpp colour depth, $m \times n \times 24$ Bits of memory are necessary. An image of 8000×6000 Pixels (4:3 aspect ratio and 48MP resolution of modern smartphone cameras) with 24 Bit colour depth will lead to an uncompressed image filesize of 144 MB. To reduce the image filesize, lossy (permanently removing *unnecessary* information of the original image) and lossless compression methods are applied. For lossy compression algorithms, it is vital to identify unnecessary information of the image, which are not perceived by the viewer and can be removed. One approach for a lossy compression is based on the Singular Value Decomposition (SVD), which will be explored in more detail in Question 2.

Question 1:

For simplicity, we assume that each pixel of a grey-scale image is considered as a real value in Question 1(a) to (e). Let $G \in {}^n\mathbb{R}^m$ be a grey-scale image of $n \times m$ pixels. A collection of image filters $F_1, F_2, \ldots, F_k \in {}^n\mathbb{R}^m$ can be used to process this image, resulting in filtered images A_l represented as:

$$A_l = F_l \circ G, \quad l = 1, 2, \ldots, k, \tag{1}$$

where $\circ$ denotes the entry-wise product. The entry-wise product (also called *Hadamard* product) of two matrices of the same size is defined as the product, where each entry of the resulting matrix is the product of the corresponding entries of the original matrices. For example, if $P = \begin{bmatrix} p_{11} & p_{12} \\ p_{21} & p_{22} \end{bmatrix}$ and $Q = \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix}$ are 2×2 matrices, the Hadamard product is defined as

$$P \circ Q = \begin{bmatrix} p_{11}q_{11} & p_{12}q_{12} \\ p_{21}q_{21} & p_{22}q_{22} \end{bmatrix}. \tag{2}$$

(a) Can the filter F_1 in Equation (1) be used to blur or smooth an image A_1? Unsupported answers will not receive full credit.

F_1 may blur/smooth an image A_1, depending on its values. Traditionally, blurring is done through "averaging filters", which assign weights to adjacent pixels and average them to reduce variations. If F_1 is derived from G itself, where the entries in F_1 are selected so that adjacent values in G average out to similar values, then it can create a blurring effect on the image (A_1). However, if F_1 is unrelated to G, this blurring cannot be achieved for all G. This is because $F_1 \circ G$ undergoes entry-wise multiplication, meaning each entry is altered independently, as opposed to an averaging filter, which considers neighboring elements during computation

(b) Name at least **three** applications in the context of digital image processing for a filter as defined in Equation (1). Additionally, give details how F_i would look like for your applications.

1) Embossing an Image: embossing is a computer graphics method that replaces each pixel of an image by either a highlight or shadow, depending on the lightness or darkness of the original image. A filter F_l such as $F_l = \begin{bmatrix} -4 & -2 & 0 \\ -2 & 1 & 3 \\ 0 & 3 & 6 \end{bmatrix}$ can give an image a raised, 3D appearance in the diagonal direction, by decreasing values at the top, and increasing those closer to the bottom.

2) Increasing brightness : To increase brightness in an image, if the entries in F_l have values greater than 1, such as $F_l = \begin{bmatrix} 1.2 & 1.2 \\ 1.2 & 1.2 \end{bmatrix}$, all entries within G would be scaled up by the same amount. This would bring values closer to 255 (white) and because this is applied for all entries, this will create an overall effect of a brighter image.

3) Reducing noise : "blurring filters" can reduce noise in images, and create a clearer output. An example of a filter F_i that could be applied is $F_l = \frac{1}{4}\begin{bmatrix} 2 & 4 & 2 \\ 4 & 16 & 4 \\ 2 & 4 & 2 \end{bmatrix}$, which gives weight to a centre pixel, while lessening that of its neighbours, to reduce noise around a pixel.

(c) Assume that the entries of the image G are all nonzero. Prove that the set of filtered images $\{F_1 \circ G, F_2 \circ G, \ldots, F_k \circ G\}$ is linearly independent if and only if the set of filters $\{F_1, F_2, \ldots, F_k\}$ is linearly independent.

($\Rightarrow$) First, to prove the if and only if, we'll prove that if $\{F_1, F_2, \ldots F_k\}$ is linearly independent (by assumption), then $\{F_1 \circ G, F_2 \circ G, \ldots F_k \circ G\}$ is linearly independent. This means for $c_1, c_2, \ldots c_k \in \mathbb{R}$, $c_1F_1 + c_2F_2 + \ldots c_kF_k = 0$ only for $c_1 = c_2 = \ldots = c_k = 0$.
To be linearly independent, $c_1(F_1 \circ G) + c_2(F_2 \circ G) + \ldots + c_k(F_k \circ G) = 0$ for $c_1, c_2, \ldots c_k \in \mathbb{R}$ and $c_1 = c_2 = \ldots c_k = 0$.
If $F_i = \begin{bmatrix} {}^ip_{11} & \cdots & {}^ip_{1m} \\ \vdots & \ddots & \vdots \\ {}^ip_{n1} & \cdots & {}^ip_{nm} \end{bmatrix}$, $G = \begin{bmatrix} g_{11} & \cdots & g_{1m} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nm} \end{bmatrix}$, this can be written as:

$$c_1 \begin{bmatrix} {}^1p_{11}g_{11} & \cdots & {}^1p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^1p_{n1}g_{n1} & \cdots & {}^1p_{nm}g_{nm} \end{bmatrix} + c_2 \begin{bmatrix} {}^2p_{11}g_{11} & \cdots & {}^2p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^2p_{n1}g_{n1} & \cdots & {}^2p_{nm}g_{nm} \end{bmatrix} + \ldots c_k \begin{bmatrix} {}^kp_{11}g_{11} & \cdots & {}^kp_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^kp_{n1}g_{n1} & \cdots & {}^kp_{nm}g_{nm} \end{bmatrix} = 0.$$

writing out the addition for each entry in the resultant matrix gives: $\sum_{i=1}^{k} c_i({}^ip_{nm})g_{nm} = 0$.
If the entries of G, being $g_{11}, g_{12}, g_{21}, g_{22}$ are all constants $\in \mathbb{R}$, and non-zero (by definition), and $c_1, c_2, \ldots c_k \in \mathbb{R}$, then this turns into:

$$\sum_{i=1}^{k} \alpha_i({}^ip_{nm}) = 0 \text{ for some } \alpha_i, \text{ where } \alpha_i = c_i \cdot g_{nm}. \quad \boxed{\text{I}}$$

However, equation $\boxed{\text{I}}$ is just k equations of linear combinations of each of the entries in $F_1, F_2, \ldots F_k$. From our assumption, $F_1, F_2, \ldots F_k$ are linearly independent, meaning all $\alpha_i = 0$. However, because $\alpha_i = c_i g_{mn}$, and g_{mn} is non-zero, then all $c_i = 0$ in $c_1(F_1 \circ G) + c_2(F_2 \circ G) + \ldots + c_n(F_n \circ G) = 0$. Because the only solution for this is that all $c_i = 0$, by definition of linear independence, $\{F_1 \circ G, F_2 \circ G, \ldots F_k \circ G\}$ is linearly independent. //

($\Leftarrow$) Now, we prove that if $\{F_1 \circ G, F_2 \circ G, \ldots F_k \circ G\}$ is linearly independent (by assumption), then so is $\{F_1, F_2, \ldots F_k\}$.
This means for $\beta_1, \beta_2, \ldots \beta_k \in \mathbb{R}$, $\beta_1(F_1 \circ G) + \beta_2(F_2 \circ G) + \ldots + \beta_k(F_k \circ G) = 0$ for $\beta_1, \beta_2, \ldots \beta_k \in \mathbb{R}$ and $\beta_1 = \beta_2 = \ldots \beta_k = 0$.
To be linearly independent, for $c_1, c_2, \ldots c_k \in \mathbb{R}$, $c_1F_1 + c_2F_2 + \ldots c_kF_k = 0$ only for $c_1 = c_2 = \ldots = c_k = 0$.
If $F_i = \begin{bmatrix} {}^ip_{11} & \cdots & {}^ip_{1m} \\ \vdots & \ddots & \vdots \\ {}^ip_{n1} & \cdots & {}^ip_{nm} \end{bmatrix}$, $G = \begin{bmatrix} g_{11} & \cdots & g_{1m} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nm} \end{bmatrix}$, this can be written as: $c_1 \begin{bmatrix} {}^1p_{11} & \cdots & {}^1p_{1m} \\ \vdots & \ddots & \vdots \\ {}^1p_{n1} & \cdots & {}^1p_{nm} \end{bmatrix} + c_2 \begin{bmatrix} {}^2p_{11} & \cdots & {}^2p_{1m} \\ \vdots & \ddots & \vdots \\ {}^2p_{n1} & \cdots & {}^2p_{nm} \end{bmatrix} + \ldots + c_k \begin{bmatrix} {}^kp_{11} & \cdots & {}^kp_{1m} \\ \vdots & \ddots & \vdots \\ {}^kp_{n1} & \cdots & {}^kp_{nm} \end{bmatrix} = 0$.
Rewriting the equations for each entry in F_i gives: $\sum c_i({}^ip_{nm}) = 0$.
Because c_i is any arbitrary constant, let $c_i = \alpha_i g_{nm}$, where $\alpha_i, g_{nm} \in \mathbb{R}$, $g_{nm} \neq 0$ (by definition, entries of G are non-zero).
This yields:

$$\sum \alpha_i g_{nm}({}^ip_{nm}) = 0.$$

This expression shows the linear combination of each entry in p_{nm} multiplied by constants α_i and g_{nm}. This is like writing $\beta_1F_1 \circ G + \beta_2F_2 \circ G + \ldots + \beta_kF_k \circ G$ for some $\beta_1, \beta_2, \ldots \beta_k \in \mathbb{R}$. However, by assumption we know that α_i from $c_i = \alpha_i g_{nm}$, has all $\alpha_i = 0$, because of the linear independence of $\{F_1 \circ G, F_2 \circ G, \ldots, F_k \circ G\}$. If all $\alpha_i = 0$, then all $c_i = 0$, because $c_i = \alpha_i g_{nm}$.
This means that only for all $c_i = 0$, $\sum_{i=0}^{k} c_i({}^ip_{nm}) = 0$ is satisfied, which, by definition of linear independence, means that $\{F_1, F_2, \ldots F_k\}$ is linear independent. //

Therefore, because both "if directions" have proven to hold, the set of images $\{F_1 \circ G, F_2 \circ G, \ldots F_k \circ G\}$ is linearly independent if and only if $\{F_1, F_2, \ldots F_k\}$ is linearly independent. //

$\therefore$ QED

(d) Assume that the entries of the image G are all nonzero. Let $\mathcal{W}$ be the set of the filtered images $\{F_1 \circ G, F_2 \circ G, \ldots, F_k \circ G\}$. Suppose a new image filter F_{k+1} is introduced. Prove that the filtered image $F_{k+1} \circ G$ lies in span $\mathcal{W}$ if and only if F_{k+1} is a linear combination of $\{F_1, F_2, \ldots, F_k\}$.

To prove this if and only if statement, we have to prove two directions:

($\Rightarrow$) first: If $F_{k+1} \circ G \subseteq \operatorname{span} W$, where $W = \{F_1 \circ G, F_2 \circ G, \ldots, F_k \circ G\}$, then F_{k+1} is a linear combination of $\{F_1, F_2, \ldots F_k\}$.

To start, if $F_{k+1} \circ G \subseteq \operatorname{span} W$, by the definition of a span, $F_{k+1} \circ G = c_1 F_1 \circ G + c_2 F_2 \circ G + \ldots + c_k F_k \circ G$, where $c_1, c_2, \ldots c_k \in \mathbb{R}$. Considering that the entries of G are all constants $\in \mathbb{R}$, for $F_i = \begin{bmatrix} {}^{i}p_{11} & \cdots & {}^{i}p_{1m} \\ \vdots & \ddots & \vdots \\ {}^{i}p_{n1} & \cdots & {}^{i}p_{nm} \end{bmatrix}$ and $G = \begin{bmatrix} g_{11} & \cdots & g_{1m} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nm} \end{bmatrix}$, this can be rewritten as:

$$\begin{bmatrix} {}^{k+1}p_{11}g_{11} & \cdots & {}^{k+1}p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^{k+1}p_{n1}g_{n1} & \cdots & {}^{k+1}p_{nm}g_{nm} \end{bmatrix} = c_1 \begin{bmatrix} {}^{1}p_{11}g_{11} & \cdots & {}^{1}p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^{1}p_{n1}g_{n1} & \cdots & {}^{1}p_{nm}g_{nm} \end{bmatrix} + c_2 \begin{bmatrix} {}^{2}p_{11}g_{11} & \cdots & {}^{2}p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^{2}p_{n1}g_{n1} & \cdots & {}^{2}p_{nm}g_{nm} \end{bmatrix} + \ldots c_k \begin{bmatrix} {}^{k}p_{11}g_{11} & \cdots & {}^{k}p_{1m}g_{1m} \\ \vdots & \ddots & \vdots \\ {}^{k}p_{n1}g_{n1} & \cdots & {}^{k}p_{nm}g_{nm} \end{bmatrix}.$$

The equation for the entries of F_{k+1} can be written as ${}^{k+1}p_{nm}g_{nm} = \sum_{i=0}^{k} c_i({}^{i}p_{nm})g_{nm}$ for some $c_i, g_{nm} \in \mathbb{R}$, $g_{nm} \neq 0$, by definition of g_{nm}.

However, because $g_{nm} \neq 0$ on each side of the equations, this can be written as: ${}^{k+1}p_{nm} = \sum_{i=0}^{k} \alpha_i \, {}^{i}p_{nm}$, $\boxed{2}$ for some $\alpha_i \in \mathbb{R}$, $\alpha_i = c_i g_{nm}$.

$\boxed{2}$ is simply the equation saying that the entries of F_{k+1} are linear combinations of those in $\{F_1, F_2, \ldots F_k\}$, as α_i are all constant coefficients. Therefore, given $F_{k+1} \circ G \subseteq \operatorname{span} W$, F_{k+1} can be written as a linear combination of $\{F_1, F_2, \ldots F_k\}$. //

($\Leftarrow$) Second: If F_{k+1} can be written as a linear combination of $\{F_1, F_2, \ldots F_k\}$, $F_{k+1} \circ G \subseteq \operatorname{span} W$.

F_{k+1} as a linear combination of $\{F_1, F_2, \ldots F_k\}$ means $F_{k+1} = c_1F_1 + c_2F_2 + \ldots c_kF_k$, for $c_1, c_2, \ldots c_k \in \mathbb{R}$. If $F_i = \begin{bmatrix} {}^{i}p_{11} & \cdots & {}^{i}p_{1m} \\ \vdots & \ddots & \vdots \\ {}^{i}p_{n1} & \cdots & {}^{i}p_{nm} \end{bmatrix}$, this can be written as:

$$\begin{bmatrix} {}^{k+1}p_{11} & \cdots & {}^{k+1}p_{1m} \\ \vdots & \ddots & \vdots \\ {}^{k+1}p_{n1} & \cdots & {}^{k+1}p_{nm} \end{bmatrix} = \begin{bmatrix} c_1 {}^{1}p_{11} & \cdots & c_1 {}^{1}p_{1m} \\ \vdots & \ddots & \vdots \\ c_1 {}^{1}p_{n1} & \cdots & c_1 {}^{1}p_{nm} \end{bmatrix} + \begin{bmatrix} c_2 {}^{2}p_{11} & \cdots & c_2 {}^{2}p_{1m} \\ \vdots & \ddots & \vdots \\ c_2 {}^{2}p_{n1} & \cdots & c_2 {}^{2}p_{nm} \end{bmatrix} + \ldots \begin{bmatrix} c_k {}^{k}p_{11} & \cdots & c_k {}^{k}p_{1m} \\ \vdots & \ddots & \vdots \\ c_k {}^{k}p_{n1} & \cdots & c_k {}^{k}p_{nm} \end{bmatrix}.$$

The equations for each of the entries in F_{k+1} look like: ${}^{k+1}p_{nm} = \sum_{i=0}^{n} c_i \, {}^{i}p_{nm}$. $\boxed{3}$

Now, the expression for $(F_{k+1}) \circ G$ written to be included in span W, with $G = \begin{bmatrix} g_{11} & \cdots & g_{1m} \\ \vdots & \ddots & \vdots \\ g_{n1} & \cdots & g_{nm} \end{bmatrix}$, $g_{nm} \in \mathbb{R}$, $g_{nm} \neq 0$ can be written as:

$$\begin{bmatrix} g_{11}({}^{k+1}p_{11}) & \cdots & g_{1m}({}^{k+1}p_{1m}) \\ \vdots & \ddots & \vdots \\ g_{n1}({}^{k+1}p_{n1}) & \cdots & g_{nm}({}^{k+1}p_{nm}) \end{bmatrix} = \alpha_1 \begin{bmatrix} g_{11}({}^{1}p_{11}) & \cdots & g_{1m}({}^{1}p_{1m}) \\ \vdots & \ddots & \vdots \\ g_{n1}({}^{1}p_{n1}) & \cdots & g_{nm}({}^{1}p_{nm}) \end{bmatrix} + \alpha_2 \begin{bmatrix} g_{11}({}^{2}p_{11}) & \cdots & g_{1m}({}^{2}p_{1m}) \\ \vdots & \ddots & \vdots \\ g_{n1}({}^{2}p_{n1}) & \cdots & g_{nm}({}^{2}p_{nm}) \end{bmatrix} + \ldots \alpha_k \begin{bmatrix} g_{11}({}^{k}p_{11}) & \cdots & g_{1m}({}^{k}p_{1m}) \\ \vdots & \ddots & \vdots \\ g_{n1}({}^{k}p_{n1}) & \cdots & g_{nm}({}^{k}p_{nm}) \end{bmatrix},$$ for $\alpha_i \in \mathbb{R}$

The equations for each of the entries in F_{k+1} look like: ${}^{k+1}p_{nm}g_{nm} = \sum_{i=0}^{n} \alpha_i g_{nm}({}^{i}p_{nm})$, where $g_{nm} \in \mathbb{R}$, $g_{nm} \neq 0$. Because the entries of g_{nm} are all non-zero, for each equation of ${}^{k+1}p_{nm}g_{nm} = \sum_{i=0}^{n} \alpha_i g_{nm}({}^{i}p_{nm})$, it can be re-written as ${}^{k+1}p_{nm} = \sum_{i=0}^{n} \alpha_i ({}^{i}p_{nm})$, when divided by g_{nm}. Therefore, we arrive at the equation:

$${}^{k+1}p_{nm} = \sum_{i=0}^{n} (\alpha_i)({}^{i}p_{nm}) \quad \boxed{4}$$

for some arbitrary $\alpha_i \in \mathbb{R}$. But, this equation, $\boxed{4}$, is equal to the one in $\boxed{3}$! α_i and c_i are both arbitrary scalar values, and by our assumption, $\boxed{3}$ is already true for F_{k+1}. This means $\boxed{4}$ must be true as well, and we know that $\boxed{4}$ is obtained by a rearrangement for the equation of $F_{k+1} \circ G \subseteq \operatorname{span} W$. Thus, if F_{k+1} can be written as a linear combination of $\{F_1, F_2, \ldots F_k\}$ ($\boxed{3}$), then $F_{k+1} \circ G \in \operatorname{span} W$ ($\boxed{4}$). //

Therefore because both directions of the if and only if statement hold, $F_{k+1} \subseteq \operatorname{span} W$ if and only if F_k is a linear combination of $\{F_1, F_2, \ldots F_k\}$. //

$\therefore$ QED. //

(e) Assume $F_1 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, $F_2 = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$, and $F_3 = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$ are filters for a grey-scale image $G \in {}^2\mathbb{R}^2$. Verify whether the filters F_1, F_2, F_3 are linearly independent, and determine whether $F_4 = \begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix}$ lies in the span of $\{F_1, F_2, F_3\}$.

To be linearly independent, $\alpha_1 F_1 + \alpha_2 F_2 + \alpha_3 F_3 = 0$ only for $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}$ and $\alpha_1 = \alpha_2 = \alpha_3 = 0$. Let's assume this set is linearly dependent, and the equation is satisfied by one non-zero α_1, α_2, or α_3. Rewriting this gives: $\begin{bmatrix} \alpha_1 & 0 \\ 0 & \alpha_1 \end{bmatrix} + \begin{bmatrix} 0 & \alpha_2 \\ \alpha_2 & 0 \end{bmatrix} + \begin{bmatrix} \alpha_3 & \alpha_3 \\ 0 & 0 \end{bmatrix} = 0$. For each entry, we obtain the equations: $\begin{cases} \alpha_1 + \alpha_3 = 0 \\ \alpha_2 + \alpha_3 = 0 \\ \alpha_1 + \alpha_2 = 0 \end{cases}$. This means $\alpha_1 = -\alpha_3$, $\alpha_1 = -\alpha_2$, and $\alpha_2 = -\alpha_3$. However, assuming one of α_1, α_2 or α_3 is non-zero, then $\alpha_1 = -(-\alpha_3) = \alpha_3$, $\alpha_1 = -(-\alpha_2) = \alpha_3$. However, this implies $\alpha_1 = \alpha_3 = -\alpha_3$. The only scalar $c \in \mathbb{R}$ that satisfies $c = -c$ is 0. But, if $\alpha_3 = 0$, then $\alpha_1 = 0$ and $\alpha_2 = 0$. We arrive at a contradiction, meaning F_1, F_2, F_3 <u>are</u> linearly independent, because only $\alpha_1 = \alpha_2 = \alpha_3 = 0$ satisfies the equation.

To determine if F_4 lies in the span of F_1, F_2, F_3, we have to find 3 scalars, $c_1, c_2, c_3 \in \mathbb{R}$, such that $F_4 = c_1 F_1 + c_2 F_2 + c_3 F_3$. This means:

$$\begin{bmatrix} 2 & 1 \\ 1 & 4 \end{bmatrix} = c_1 \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} + c_2 \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} + c_3 \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}.$$

Writing an equation for each entry, we get 4 equations: $\begin{cases} 2 = c_1 + c_3 \\ 1 = c_2 + c_3 \\ 1 = c_2 \\ 4 = c_1 \end{cases}$ $\rightarrow$ If $c_2 = 1$, $c_1 = 4$, then $c_3 = 0$, $c_3 = -2$. This implies $c_3 = 0$ and $c_3 = -2$, implying $0 = -2$, which is <u>not</u> true. Therefore, because F_4 cannot be obtained by linear factors of F_1, F_2, F_3, it does <u>not</u> lie in the span of $\{F_1, F_2, F_3\}$. //

$\therefore$ QED.

(f) For saving grey-scale images A of size $n \times m$ pixels in an efficient way, each pixel is stored as an 8 Bit integer, where 0 represents black and 255 represents white (see preamble). Let $\mathcal{V}$ be the set of these grey-scale images. Is $\mathcal{V}$ a vector space if one uses entry-wise vector addition and scalar multiplication?

<u>No</u>. If the entries of A are all supposed to be more than 0 (no negative entries), then V does not satisfy MI: that for every $\alpha \in \mathbb{R}$, $\vec{v} \in V$, $\alpha\vec{v} \in V$. If $\alpha < 0$, then matrix A would have negative entries, which by definition, don't correspond to any value on the greyscale. Therefore, because not <u>all</u> 8 axioms hold, V is not a vector space. //

$\therefore$ QED

Question 2:

Let $A \in {}^m\mathbb{R}^n$ be a grey-scale image of size $m \times n$, where each entry is representing a grey pixel. As you learnt in ESC103, the rank of an $n \times m$ matrix is the number of linear independent rows, which is the same as the number of the linear independent columns.

The Singular Value Decomposition[1] of a matrix A is a way of writing A as the product of three matrices:

$$A = U\Sigma V^T \tag{3}$$

with the matrices U, Σ, and V^T defined as

$$A = \begin{bmatrix} \mathbf{u_1} & \mathbf{u_2} & \cdots & \mathbf{u_{n-1}} & \mathbf{u_r} \end{bmatrix} \begin{bmatrix} \sigma_1 & & \cdots & & 0 \\ & \sigma_2 & & & \\ \vdots & & \ddots & & \vdots \\ & & & \sigma_{r-1} & \\ 0 & & \cdots & & \sigma_r \end{bmatrix} \begin{bmatrix} \mathbf{v_1}^T \\ \mathbf{v_2}^T \\ \vdots \\ \mathbf{v_{r-1}}^T \\ \mathbf{v_r}^T \end{bmatrix} \tag{4}$$

where $\mathbf{u}_i$ are the columns of U and $\mathbf{v}_i^T$ are the rows of V^T. The matrix Σ is a diagonal matrix with the entries σ_i on the diagonal (the diagonal entries of Σ are nonnegative and all other entries of Σ are zero). Because Σ is diagonal, Equation (4) can be written as:

$$A = \mathbf{u_1}\sigma_1\mathbf{v_1}^T + \mathbf{u_2}\sigma_2\mathbf{v_2}^T + \cdots + \mathbf{u_{r-1}}\sigma_{r-1}\mathbf{v_{r-1}}^T + \mathbf{u_r}\sigma_r\mathbf{v_r}^T. \tag{5}$$

As you can see, A is separated into the sum of (rank=1) matrices of the form $\sigma_i\mathbf{u_i}\mathbf{v_i}^T$. With that, the sum in Equation (5) can be expressed as

$$A = \sum_{i=1}^{r} \sigma_i \mathbf{u_i}\mathbf{v_i}^T \tag{6}$$

with r so called singular values $\{\sigma_i \in \mathbb{R} \mid \sigma_i > 0\}$ and both $\{\mathbf{u}_1, \ldots \mathbf{u}_r\} \subseteq {}^m\mathbb{R}$ and $\{\mathbf{v}_1, \ldots \mathbf{v}_r\} \subseteq {}^n\mathbb{R}$ are orthonormal sets of vectors.

What's an "orthonormal set of vectors"? In simple language, each vector has length one and any two vectors are perpendicular (also called "orthogonal"). More specifically, two vectors, $\mathbf{x}, \mathbf{y} \in {}^n\mathbb{R}$, are called *orthogonal* if and only if their dot-product is equal to zero: $\mathbf{x} \cdot \mathbf{y} = \mathbf{x}^T\mathbf{y} = 0$. The set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_k\} \subseteq {}^n\mathbb{R}$ is an *orthogonal set of vectors* if and only if every pair of vectors is orthogonal: $\mathbf{x}_i \cdot \mathbf{x}_j = 0$ if $i \neq j$. The set $\{\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_k\} \subseteq {}^n\mathbb{R}$ is an *orthonormal set of vectors* if and only if every pair of vectors is orthogonal and each vector has length 1: $\mathbf{x}_i \cdot \mathbf{x}_j = 0$ if $i \neq j$ and $\mathbf{x}_i \cdot \mathbf{x}_i = 1$ for every $1 \leq i, j \leq k$.

At the end of this course, you will have the tools you need to be able to find the SVD of a general matrix.[2] We can use Equation (5) to compress the image A by keeping the "most relevant" terms in the sum: As you can see from Equation (5), some matrices $\sigma_i\mathbf{u_i}\mathbf{v_i}^T$ could be neglected if σ_i is very small. Therefore, a typical strategy for compressing images is putting the singular values σ_i in nonincreasing order, $\sigma_1 \geq \sigma_2 \geq \ldots \sigma_r > 0$ (as well as sorting the corresponding $\mathbf{u}_i$ and $\mathbf{v}_i$ in U and V, respectively) and only storing $k < r$ of these components in an image file (k is an integer of at least 1). The compressed image A_k is constructed as

$$A_k = \sum_{i=1}^{k} \sigma_i \mathbf{u_i}\mathbf{v_i}^T \tag{7}$$

This will reduce the image size dramatically, however, the image quality will be degraded if k was picked too small, see for example Figure 2 where 30 components were used ($k = 30$) for the compression of the initial grey-scale image.

[1] In this writing assignment, we are introducing you to the compact (or reduced) SVD, which is slightly different from the full SVD. To save space, we refer to it as the "SVD" throughout.

[2] *For the curious:* $\mathbf{u}_i$ are eigenvectors of the matrix AA^T and $\mathbf{v}_i$ are eigenvectors of the matrix A^TA.

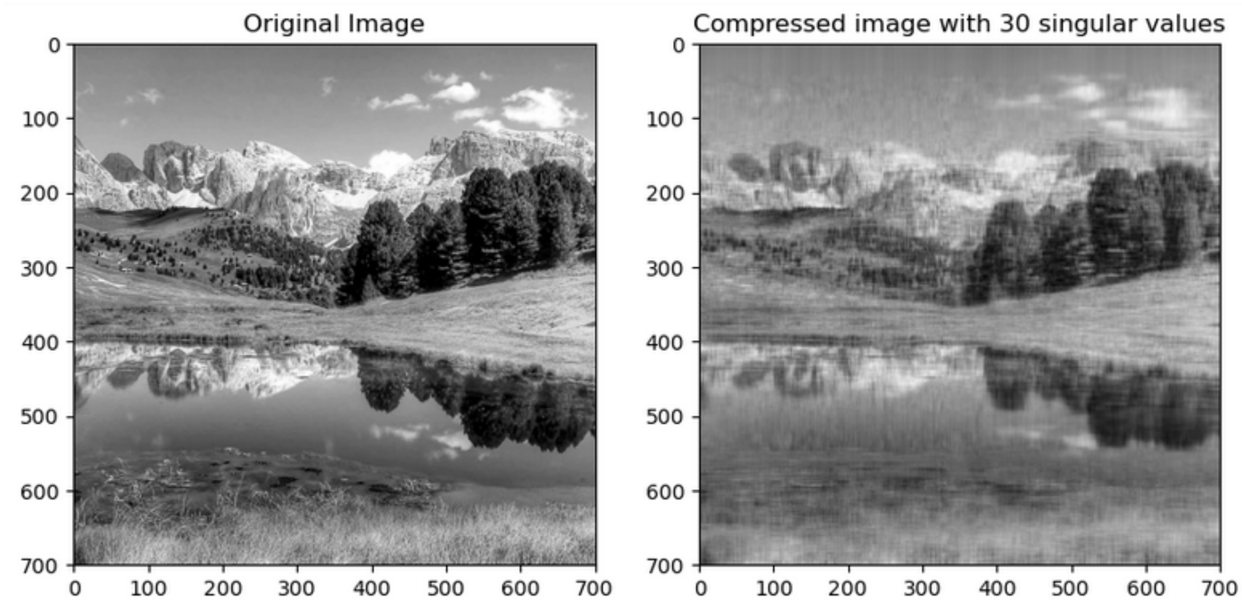


Figure 2: Lossy compression of a 700×700 pixel grey-scale image via Single Value Decomposition (SVD).

(a) For what values of k would A_k corresponds to a lossy compressed image? No additional explanation is necessary.

$$k < r$$

(b) Let $A = \sigma \mathbf{x}\mathbf{y}^T$ where $\mathbf{x}, \mathbf{y} \in {}^n\mathbb{R}$ and $\sigma \in \mathbb{R}$ is nonzero. Prove that $\text{rank}(A) = 1$. *Hint:* Try constructing a 3×3 example, A, by choosing some nonzero $\sigma \in \mathbb{R}$ and $\mathbf{x}, \mathbf{y} \in {}^3\mathbb{R}$. This should give you the insight you need to answer this question for general $\mathbf{x}, \mathbf{y} \in {}^n\mathbb{R}$ below.

Since $\sigma \in \mathbb{R}$ is non-zero, and a scalar value, $A = \sigma x y^T$ can be written as:

$$A = \sigma \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \begin{bmatrix} y_1^T & y_2^T & \cdots & y_n^T \end{bmatrix} = \sigma \begin{bmatrix} x_1 y_1^T & x_1 y_2^T & \cdots & x_1 y_n^T \\ x_2 y_1^T & x_2 y_2^T & \cdots & x_2 y_n^T \\ \vdots & \vdots & \ddots & \\ x_n y_1^T & x_n y_2^T & & x_n y_n^T \end{bmatrix}$$

As seen, the result is an $n \times n$ matrix, where each column is a linear combination of $\begin{bmatrix} x_1 y_1^T \\ x_2 y_1^T \\ \vdots \\ x_n y_1^T \end{bmatrix}$

The first column can be obtained by multiplying by 1, the second can be obtained by taking $\frac{y_2^T}{y_1^T}$ where $y_2^T, y_1^T \in \mathbb{R}$, the third by $\frac{y_3^T}{y_1^T}$ where $y_3^T, y_1^T \in \mathbb{R}$, and so on, until y_n^T.

Since the $n \times n$ matrix is composed of linear combinations of $\begin{bmatrix} x_1 y_1^T \\ x_2 y_1^T \\ \vdots \\ x_n y_1^T \end{bmatrix}$, the rank of this matrix is 1.

Since $\sigma \in \mathbb{R}$ is nonzero, and multiplying the $n \times n$ matrix by a non-zero scalar value doesn't change its rank, the rank of $A = \sigma x y^T$ is 1. Thus, $\text{rank}(A) = 1$. ∎

(c) Let $A = \begin{bmatrix} 36 & 9 & 12 \\ -48 & -12 & -16 \\ 144 & 36 & 48 \end{bmatrix}$. Determine the singular value decomposition $A = \sigma \mathbf{x}\mathbf{y}^T$.

Since $\begin{bmatrix} 36 & 9 & 12 \\ -48 & -12 & -16 \\ 144 & 36 & 48 \end{bmatrix} = (1)\begin{bmatrix} 36 & 9 & 12 \\ -48 & -12 & -16 \\ 144 & 36 & 48 \end{bmatrix} = (1)\begin{bmatrix} 3 \\ -4 \\ 12 \end{bmatrix}\begin{bmatrix} 12 & 3 & 4 \end{bmatrix} = (1)\dfrac{\begin{bmatrix} 3 \\ -4 \\ 12 \end{bmatrix}}{\sqrt{3^2+(-4)^2+12^2}} \dfrac{\begin{bmatrix} 12 & 3 & 4 \end{bmatrix}}{\sqrt{12^2+3^2+4^2}} = (169)\begin{bmatrix} 3/13 \\ -4/13 \\ 12/13 \end{bmatrix}\begin{bmatrix} \frac{12}{13} & \frac{3}{13} & \frac{4}{13} \end{bmatrix}$

This matrix is now in the form $A = \sigma x y^T$ with $\sigma = 169$, $x = \begin{bmatrix} 3/13 \\ -4/13 \\ 12/13 \end{bmatrix}$, $y^T = \begin{bmatrix} \frac{12}{13} & \frac{3}{13} & \frac{4}{13} \end{bmatrix}$

Using the given equation (5), the singular value decomposition of $A = (169)(\frac{3}{13})(\frac{12}{13}) + (169)(\frac{-4}{13})(\frac{3}{13}) + (169)(\frac{12}{13})(\frac{4}{13}) = 72$

Therefore, the sin ular value decomposition of A is 72.

(d) Assume $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ is an orthogonal set of nonzero vectors. Prove that the set is linearly independent.

Since V_1, V_2, V_3 is an orthogonal set and are non-zero vectors, $V_1 \cdot V_2 = 0$, $V_2 \cdot V_3 = 0$, $V_1 \cdot V_3 = 0$.

Let $\alpha_1, \alpha_2, \alpha_3 \in \mathbb{R}$ be scalar values such that $\alpha_1 V_1 + \alpha_2 V_2 + \alpha_3 V_3 = 0$.

$$\alpha_1 V_1 + \alpha_2 V_2 + \alpha_3 V_3 = 0$$
$$(\alpha_1 V_1 + \alpha_2 V_2 + \alpha_3 V_3) \cdot V_i = 0 \cdot V_i \quad \text{where } i = 1, 2, \text{ or } 3 \text{ (meaning } V_1, V_2, \text{ or } V_3)$$
$$\alpha_1 V_1 \cdot V_i + \alpha_2 V_2 \cdot V_i + \alpha_3 V_3 \cdot V_i = 0$$
$$\alpha_1 (V_1 \cdot V_i) + \alpha_2 (V_2 \cdot V_i) + \alpha_3 (V_3 \cdot V_i) = 0$$

Since if $V_j \neq V_i$ where $j = 1, 2$, or 3 (V_1, V_2, or V_3), then $V_j \cdot V_i = 0$ since it's an orthogonal set of vectors. This leaves if $V_j = V_i$. If $V_j = V_i$, then:

$$\alpha_i (V_i \cdot V_i) = 0$$
$$\alpha_i (V_i V_i^T) = 0$$

Since $V_i \neq 0$, then $V_i V_i^T \neq 0$. Thus to satisfy the equation, $\alpha_i = 0$.

Thus, $\alpha_1 = \alpha_2 = \alpha_3 = 0$, meaning the set must be linearly independent. ∎

(e) Consider the set $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3\}$ from part (d). What property of a 4th vector $\mathbf{v}_4 \in {}^4\mathbb{R}$ would ensure that the extended set $\{\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4\}$ spans ${}^4\mathbb{R}$? Give a short explanation. Note: your explanation does not have to include how you would find such a $\mathbf{v}_4$.

The property that V_4 is orthogonal and linearly independent to V_1, V_2, V_3 would ensure that $\{V_1, V_2, V_3, V_4\}$ spans ${}^4\mathbb{R}$.

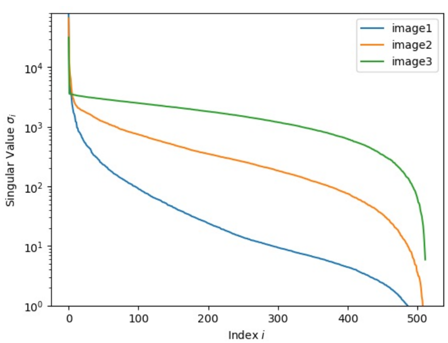
From part d, $\{V_1, V_2, V_3\}$ is an linearly independent and orthogonal set, similar to the x-y-z planes which are orthogonal to each other. This makes it so that any point or vector on $\mathbb{R}^3$ can be found. Due to this, 3 linearly independent and orthogonal vectors are needed to span ${}^3\mathbb{R}$.

Taking this logic, to span ${}^4\mathbb{R}$, you would need another vector that is orthogonal and linearly independent from the other vectors in the set. Therefore, for the set $\{V_1, V_2, V_3, V_4\}$ to span ${}^4\mathbb{R}$, V_4 would have to be linearly independent from V_1, V_2, V_3 and orthogonal to V_1, V_2, V_3.

(f) Download the Jupyter notebook (see Quercus page of *Assignment 2*), which includes the python code of an SVD analysis. Please also download the three example grey-scale images *image1.png*, *image2.png*, and *image3.png* from the same Quercus page. You can run the code on the U of T Jupyter server, on your own local Jupyter installation, or on the ECF lab PCs. We recommend using the U of T Jupyter server, since all necessary packages are already installed: https://jupyter.utoronto.ca/hub/user-redirect/tree. For that, please login with your U of T credentials and click '*upload*' to upload the IPYNB file and images. After that, you can open the uploaded notebook and run the Python code.

Use the Jupyter Python code to analyze the provided three images. Plot the sorted singular values σ_i over the index i in one graph and discuss the results for all three provided images. Do you see a pattern of the result depending on each input image?

Insert your plot below. In addition, briefly discuss which of the 3 images would allow for the highest and which for the lowest compression ratio by singular value decomposition. *Hint:* The compression ratio is the uncompressed file size over the compressed file size. You don't need to calculate this ratio. However, think about which image would need the largest k and which the lowest k in Equation (7) for an accurate representation of the uncompressed image, Equation (6). For plotting σ_i, see the comments and plotting commands in the Jupyter file. Also, keep the logarithmic Y-axis for your graph.



Since the y-axis shows the value of each value of σ_i at the index i indicated in the x-axis. Since K is in the denominator, a smaller K value results in a larger compression ratio.

Since the value of each σ_i for image 1 decreases most drastically at smaller indices, thus image 1 would need the lowest K for an accurate representation of the uncompressed image.

Thus, image 1 would allow for the highest compression ratio

Since the value of each σ_i for image 3 only decreases at the end (value stays relatively the same throughout the indices seen through the plateau)

Therefore, image 3 would need the highest K for an accurate representation of the uncompressed image

Thus image 3 would allow for the smallest compression ratio

(g) Calculate the SVD of *image1.png* with the given Jupyter code. For what choice of $k \in \mathbb{Z}$ would you expect a reduction in file size? To answer this question, estimate the memory needed to store the SVD-compressed image and compare it to the memory needed for the uncompressed 8 bpp version of the image. *Hint:* Estimate the memory size of the uncompressed image by using its size and given colour depth. Also note that each floating-point number uses 32 Bit memory when stored.

Since the image is 512×512 pixels, and the uncompressed version is 8bpp.

Therefore, the memory for the uncompressed image is $(512)(512)(8) = 2097152$ bits

To calculate the choice of K for a reduction in file size, the following equation was solved, where $(32)(512K + K + 512K)$ represents the memory of the SVD-compressed image storing K components.

$$(8)(512)(512) = (32)(512K + K + 512K)$$

$$\frac{(8)(512)(512)}{32} = 1025K$$

$K \doteq 63.94 \doteq 63$ (rounded down since only $K \leq 63.94$ will result in a smaller memory)

Thus, I would only expect a reduction in file size if $K \leq 63$.