APPLIED LINEAR ALGEBRA (MAT3004)

USE OF LINEAR ALGEBRA IN APPLICATION:

In one of the triannual SIAM conferences on applied linear algebra, a diverse group of internationally recognized scientific corporations and government laboratories was asked how linear algebra finds application in their missions.

The overwhelming response was that the primary use of linear algebra in applied industrial and laboratory work involves the development, analysis, and implementation of numerical algorithms along with some discrete and statistical modelling.

APPLICATION AREA:

- Image Processing
- Flow in a network of pipes
- Current and voltage in LCR circuits
- Cryptography
- Machine learning (Support Vector Machine, Principle Component Analysis)
- Load and displacements in structures
- Finite element analysis (has Mechanical, Electrical, and Thermodynamic applications)
- Many more…

WHY WE STUDY LINEAR ALGEBRA?

- One of the primary objective of the linear algebra course is to study a new necessary and sufficient conditions to solve the system of linear equations .

Consider the following system of linear equations,

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2$$
$$\dots$$
$$\dots$$
$$a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_n$$

WHY WE STUDY LINEAR ALGEBRA?

- The system we have seen in the last slide can be written in matrix form $AX = B$ Where $A = (a_{ij})$ is the coefficient $m \times n$ matrix, X is the $n \times 1$ matrix consisting of all variables and B is the $m \times 1$ matrix consists the right hand side of the system.
- We know if $m = n \; and \; m, n \leq 3$ The following are equivalent

(i) Determinant of $A \neq 0$.

(ii) System $AX = B$ has a unique solution.

WHY WE STUDY LINEAR ALGEBRA?

- If $m \neq n, m, n \geq 3$ then the system may or may not have solution. The following question arise naturally,

Question 1 : Do we have any specific methods to solve the system of linear equations of the above mentioned type?

Question 2: Can we derive some equivalent conditions for the system has either no solution or have infinitely many solution or have unique solution based on the nature of A?

WHY WE STUDY LINEAR ALGEBRA?

We answer to the above question by studying this linear algebra course.

MODULE 1:

- Foucues on the following techniques to solve the system of linear equations

1) Gauss Elimination method (Very general method to solve any system of linear equations)
2) Gauss Jordan Elimination method (It is very helpful to solve the system $AX = B$ when A is invertible)
3) LU decomposition method (It is very helpful to solve the system $AX = B$ in a very effective manner when A is fixed and B varies)

System of Linear Equations:

Gaussian elimination and Gauss Jordan methods - Elementary matrices- permutation matrix - inverse matrices - System of linear equations - - LU factorizations.

MODULE 2,3,4

- Focuses on the question ``deriving some equivalent conditions for the system AX=B has either no solution or have infinitely many solution or have unique solution based on the nature of A.

Vector Spaces:

The Euclidean space R^n and vector space- sub space –linear combination- span-linearly dependent-independent- bases - dimensions-finite dimensional vector space.

Subspace Properties:

Row and column spaces -Rank and nullity – Bases for subspace – invertibility- Application in interpolation.

Linear Transformations and applications:

Linear transformations – Basic properties-invertible linear transformation - matrices of linear transformations - vector space of linear transformations – change of bases – similarity

MODULE 5,6

- Focuses on developing tools to get a approximate solutions of the inconsistent system AX=B.
- And also these modules help us to understand the concept of Singular Value Decomposition which is applicable in Digital Image Processing.

Inner Product Spaces:

Dot products and inner products – the lengths and angles of vectors – matrix representations of inner products- Gram-Schmidt orthogonalization

Applications of Inner Product Spaces:

QR factorization- Projection - orthogonal projections – relations of fundamental subspaces –Least Square solutions in Computer Codes

MODULE 7

- We learn application of linear algebra in CRYPTOGRAPHY AND WAVELET ANALYSIS.

Applications of Linear equations :

An Introduction to coding - Classical Cryptosystems –Plain Text, Cipher Text, Encryption, Decryption and Introduction to Wavelets (only approximation of Wavelet from Raw data)

IMAGE COMPRESSION AND LINEAR ALGEBRA

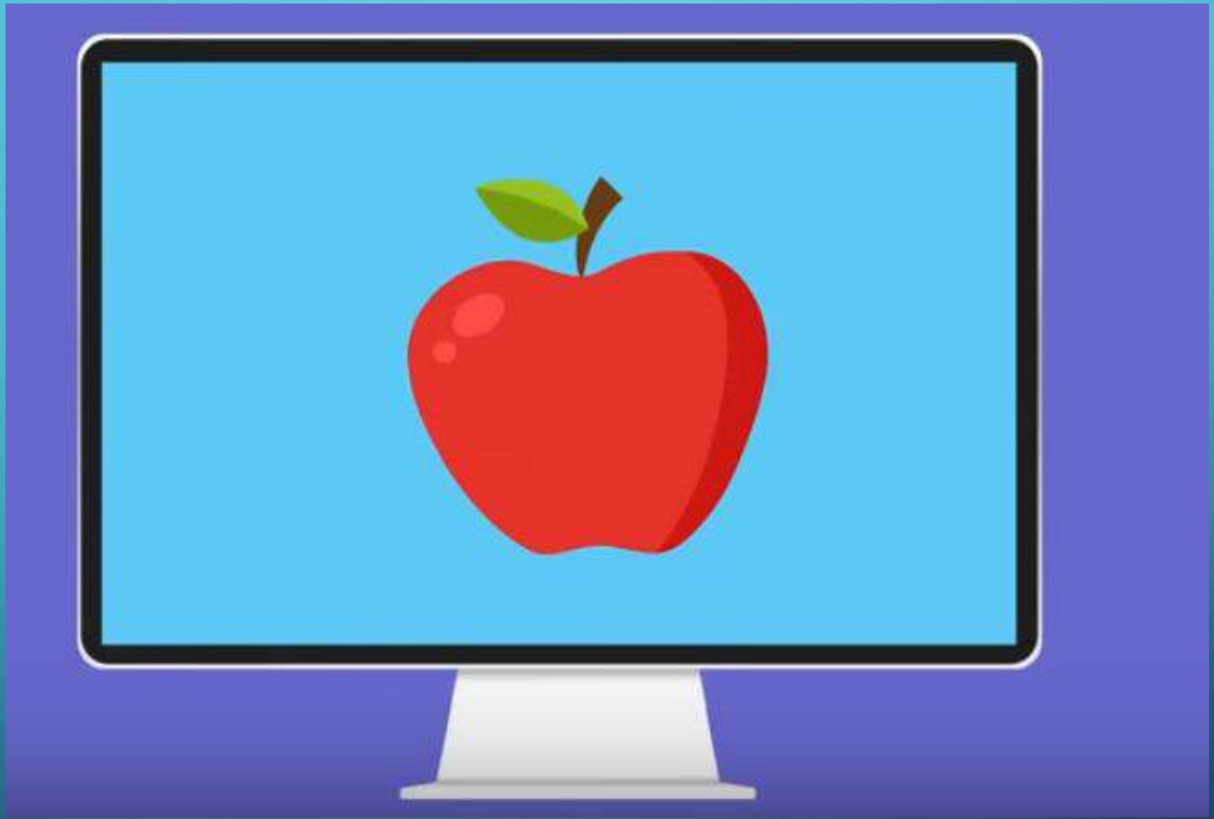
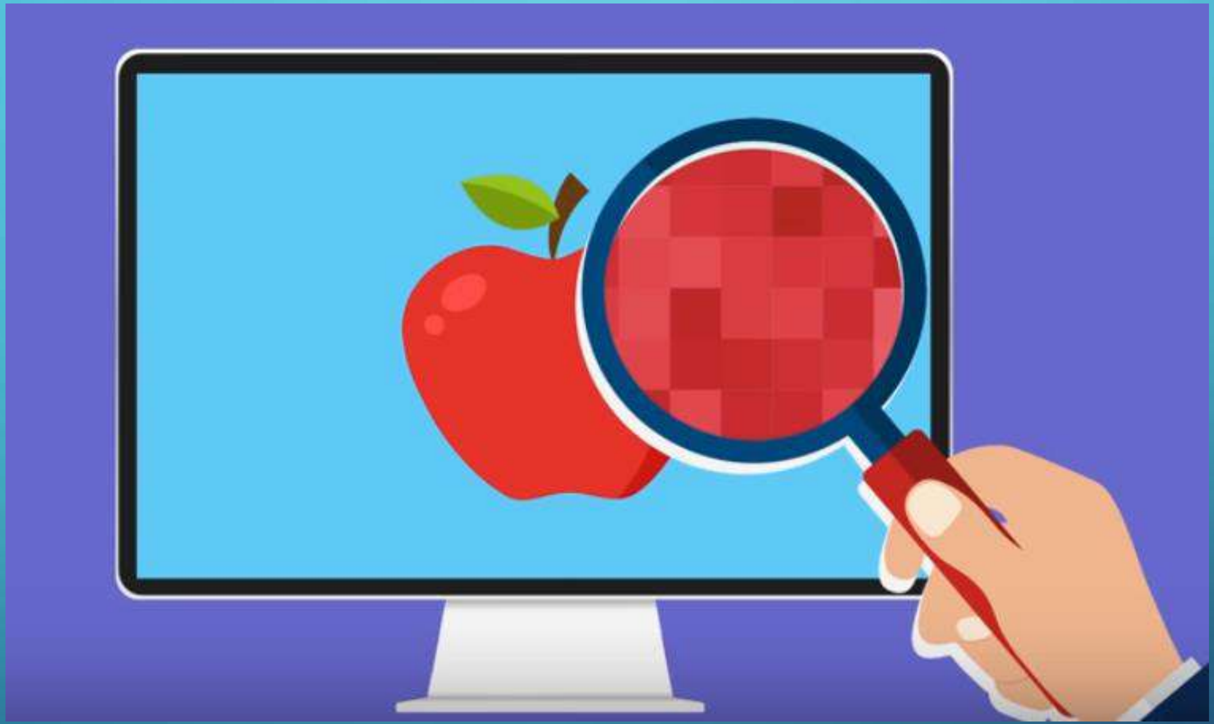


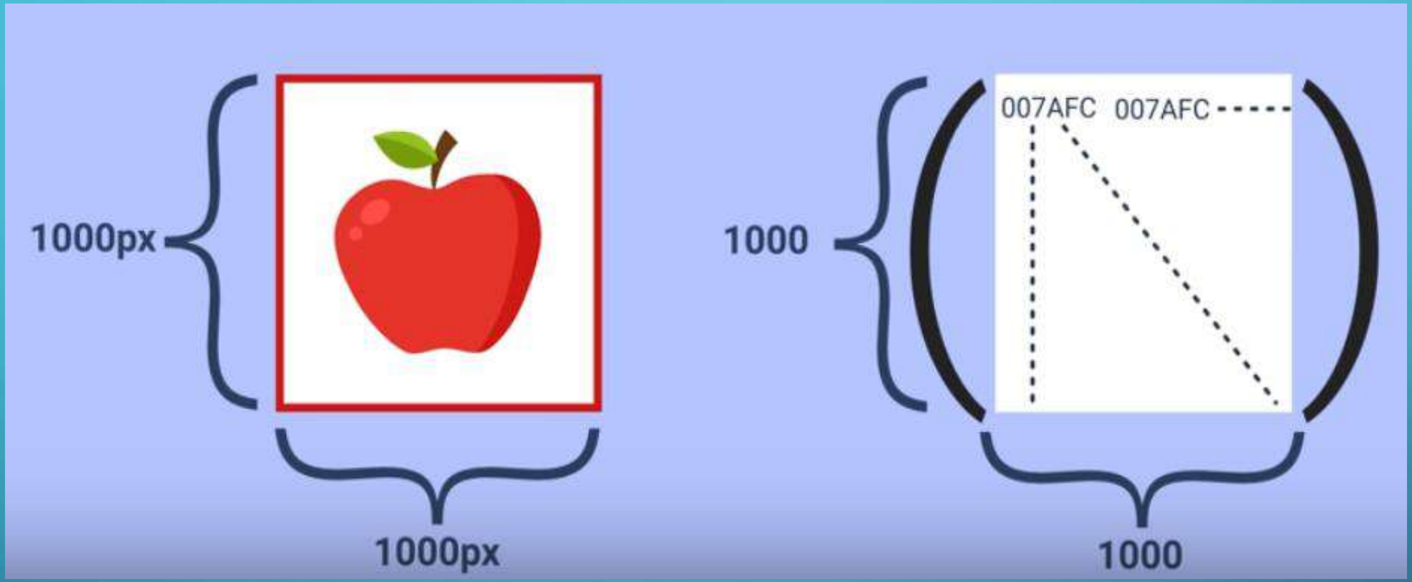
IMAGE COMPRESSION AND LINEAR ALGEBRA



- In a digital image if you look very closely (zoom in) is just made up of bunch of pixels each of a single colors. Those colors are represented by some numerical values

IMAGE COMPRESSION AND LINEAR ALGEBRA

IMAGE COMPRESSION AND LINEAR ALGEBRA



- Each side has 1000 pixels. So, the square picture made of a million pixels.
- That can be represented by 1000 x 1000 matrix, where the entries are the color value of each pixel.

IMAGE COMPRESSION AND LINEAR ALGEBRA

Pixel count

In order to calculate this resolution you just use the same formula you would use for the area of any rectangle multiply the length by the height.

For example, if you have a photo that has 4,500 pixels on the horizontal side, and 3,000 on the vertical size it gives you a total of 13,500,000.

Because this number is very unpractical to use, you can just divide it by a million to convert it into megapixels. So 13,500,000 / 1,000000 = 13.5 Megapixels.

IMAGE COMPRESSION AND LINEAR ALGEBRA

- An mxn pixels image can be represented by mxn matrix representation.
- Suppose we have an 9 megapixel, gray-scale image, which is 3000×3000 pixels (a 3000×3000 matrix).
- For each pixel, we have some level of black and white color, given by some integer between 0 and 255.0 representing black color and 255 representing white color.

IMAGE COMPRESSION AND LINEAR ALGEBRA

The basic idea here is each image can be represented as a matrix and we apply linear algebra (Singular Value Decomposition) on this matrix and get a reduced matrix out of this original matrix and the image corresponding to this reduced matrix requires much lesser storage space as compared to the original image.

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Any real matrix A can be factored as $A = U\Sigma V^T$ where U and V are orthogonal and Σ is diagonal with diagonal elements $\sigma_i \geq 0$ (called singular values of A).

IMAGE COMPRESSION AND LINEAR ALGEBRA

Any real $m \times n$ matrix A can be expressed as a finite sum of rank 1 matrices in normalized form, that is $A = \sigma_1 R_1 + \sigma_2 R_2 + \cdots + \sigma_k R_k,$ where $k = \min(m, n)$ and

1)$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_r > 0 = \sigma_{r+1} = \sigma_{r+2} = \cdots = \sigma_n, rank(A) = r \leq k.$

2)$R_i = p_i q_i^T$ where p_i is the i^{th} column of P and a unit eigenvector of AA^T an q_i is the i^{th} column of Q and a unit eigenvector of $A^T A$.

3)Each R_i has the sum of squares of its elements equal to 1.

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Let $A = U\Sigma V^T$ be the singular value decomposition of A. If $k < r = rank(A)$ and $A_k = \sum_{i=1}^{k} \sigma_i p_i q_i^T$ then

$$\min_{rank(B)=k} \|A - B\|_2 = \|A - A_k\|_2 = \sigma_{k+1}.$$

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Consider the following image

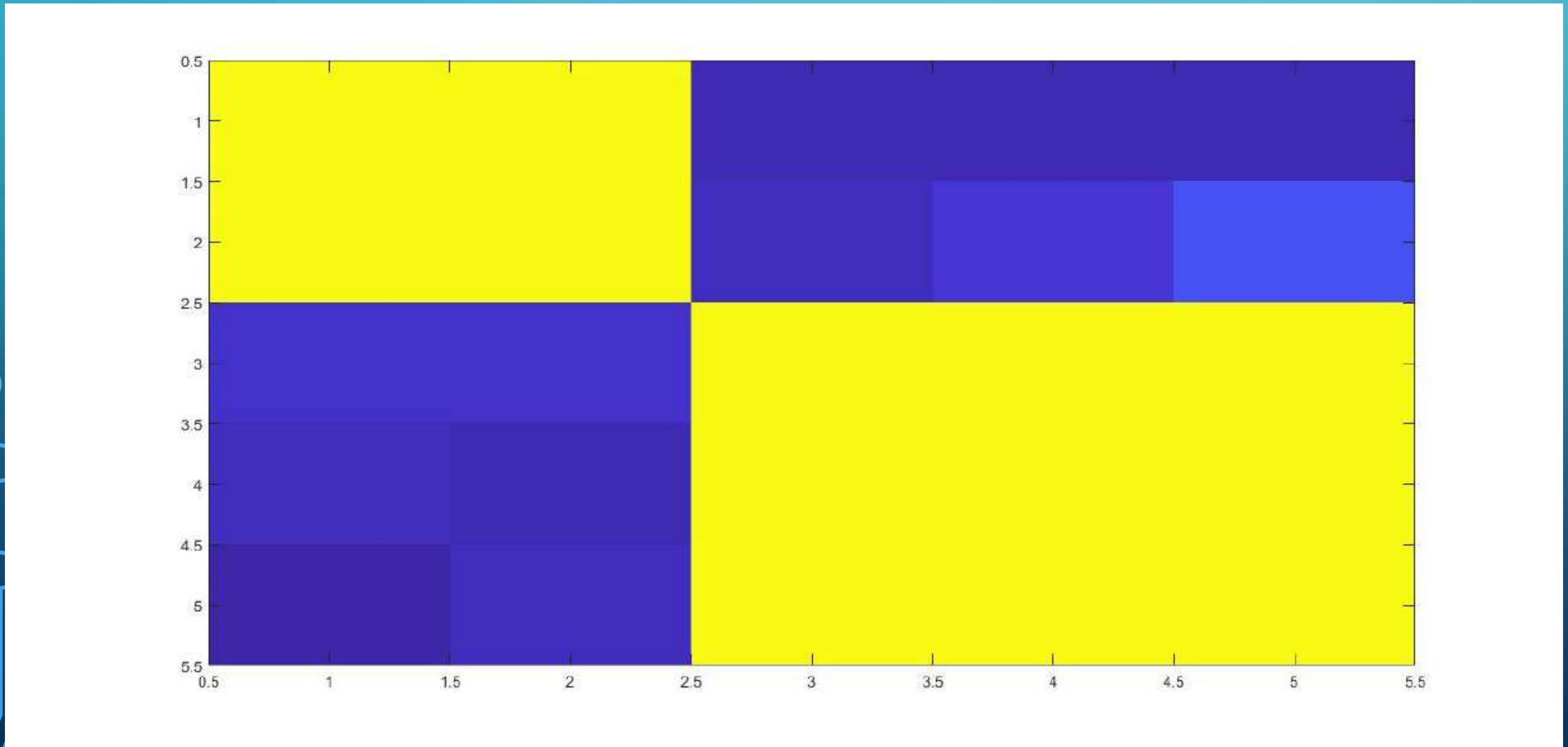


IMAGE COMPRESSION AND LINEAR ALGEBRA

- Matrix corresponds to the above image is
- $$\begin{pmatrix} 255 & 255 & 2 & 2 & 2 \\ 255 & 255 & 3 & 5 & 10 \\ 4 & 4 & 255 & 253 & 255 \\ 3 & 2 & 255 & 255 & 255 \\ 1 & 3 & 255 & 255 & 254 \end{pmatrix}$$

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Singular value decomposition is

U =

-0.0195	0.7069	0.6800	-0.1915	0.0298
-0.0286	0.7065	-0.6786	0.1959	-0.0333
-0.5763	-0.0164	-0.1637	-0.4836	0.6379
-0.5777	-0.0206	-0.0455	-0.2964	-0.7589
-0.5770	-0.0220	0.2196	0.7767	0.1233

IMAGE COMPRESSION AND LINEAR ALGEBRA

$$S = \begin{bmatrix} 764.2936 & 0 & 0 & 0 & 0 \\ 0 & 509.7433 & 0 & 0 & 0 \\ 0 & 0 & 3.7256 & 0 & 0 \\ 0 & 0 & 0 & 1.6245 & 0 \\ 0 & 0 & 0 & 0 & 1.2135 \end{bmatrix}$$

IMAGE COMPRESSION AND LINEAR ALGEBRA

V =

-0.0221	0.7068	-0.0601	-0.5699	-0.4142
-0.0228	0.7067	0.0700	0.5687	0.4144
-0.5777	-0.0225	0.5366	-0.4008	0.4660
-0.5763	-0.0197	0.2602	0.4358	-0.6402
-0.5772	-0.0128	-0.7974	-0.0346	0.1723

IMAGE COMPRESSION AND LINEAR ALGEBRA

One can check that

$$A = USV^T.$$

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Consider the following new matrix

S1 =

$$\begin{bmatrix} 764.2936 & 0 & 0 & 0 & 0 \\ 0 & 509.7433 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Matrix of US_1V^T is

254.9898	254.9846	0.4989	1.4995	4.0030
255.0128	255.0128	4.5032	5.4933	8.0020
3.8363	4.1687	254.6516	253.9966	254.3532
2.3339	2.6673	255.3270	254.6643	255.0069
1.8302	2.1632	254.9968	254.3331	254.6703

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Image of US_1V^T is

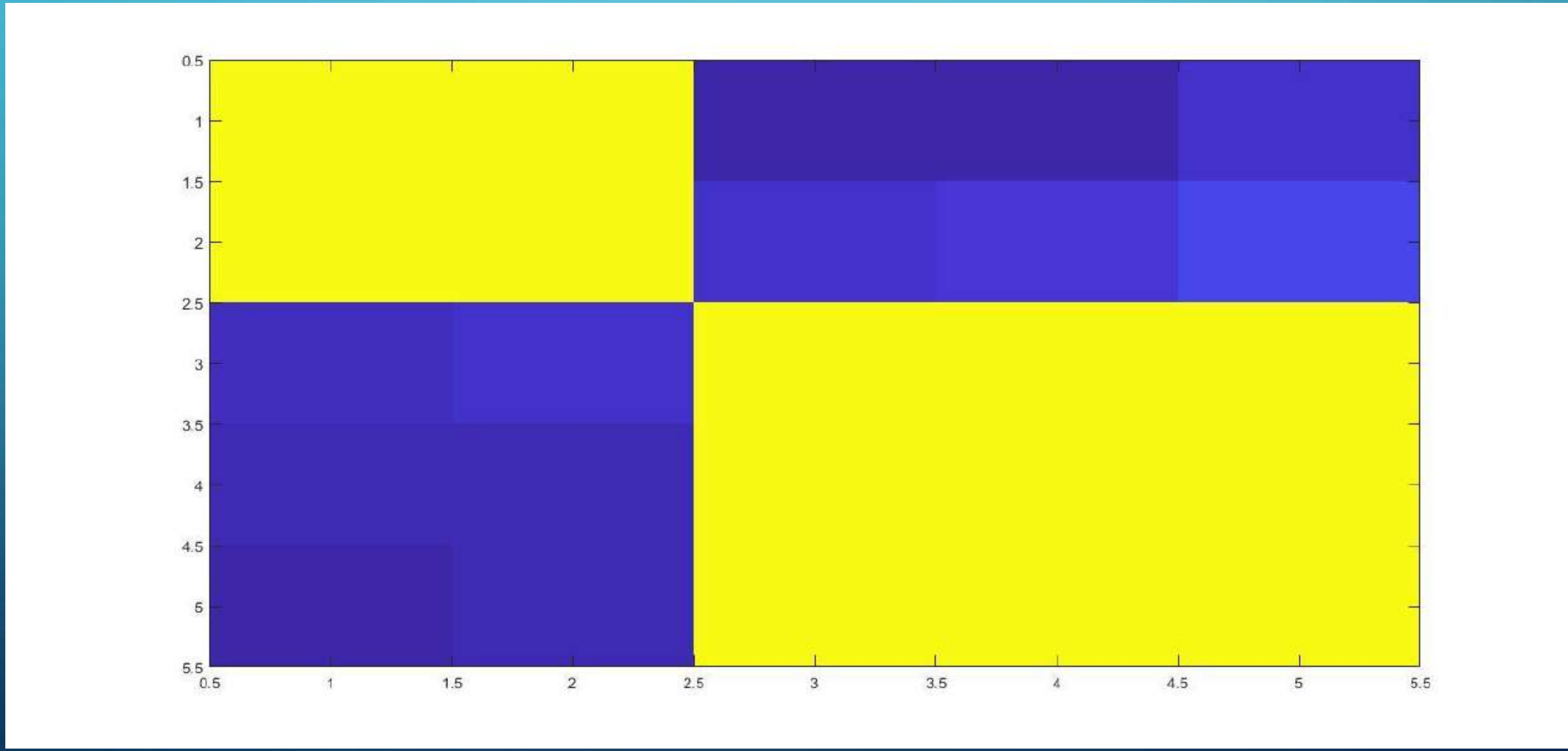


IMAGE COMPRESSION AND LINEAR ALGEBRA

The image USV^T and US_1V^T looks almost same.

IMAGE COMPRESSION AND LINEAR ALGEBRA

- Consider the following 340×280 pixel image.

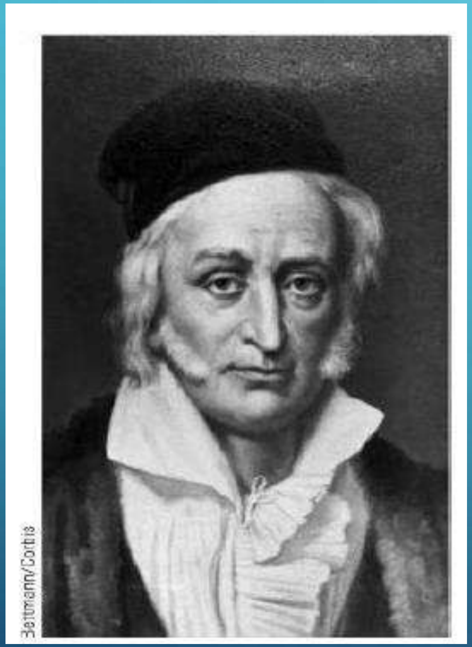


IMAGE COMPRESSION AND LINEAR ALGEBRA

In order to store the above image, we need to store 95200 numbers. In this case transmission and manipulation of this image with 95200 numbers is very expensive.

So we apply singular value decomposition for compress the image resolution and without losing the important information which is available in the image.

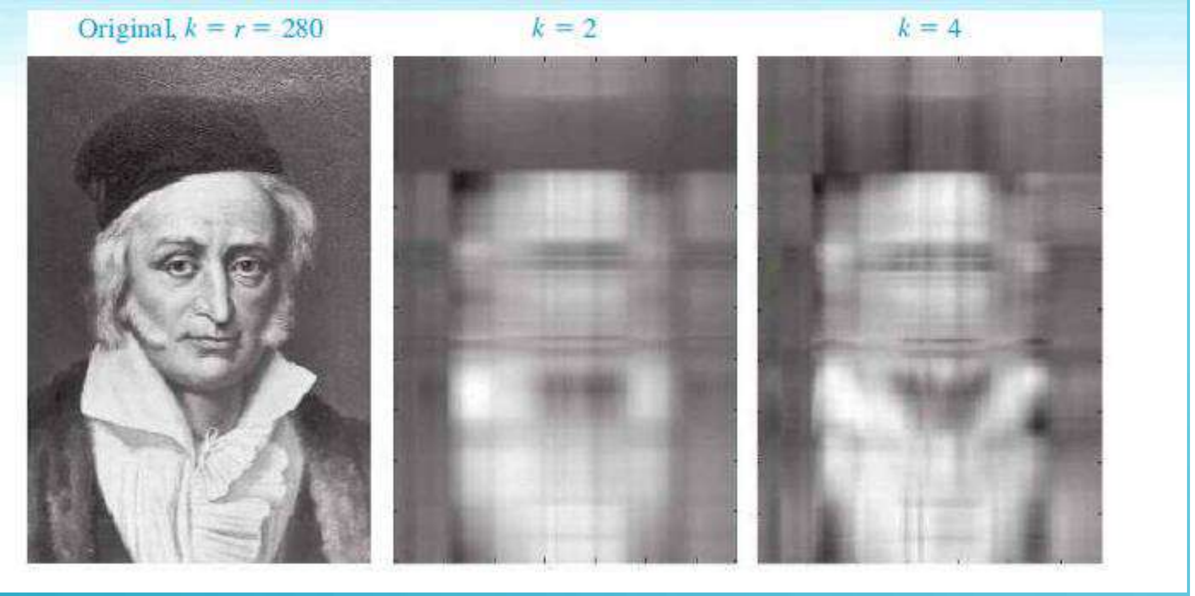
IMAGE COMPRESSION AND LINEAR ALGEBRA

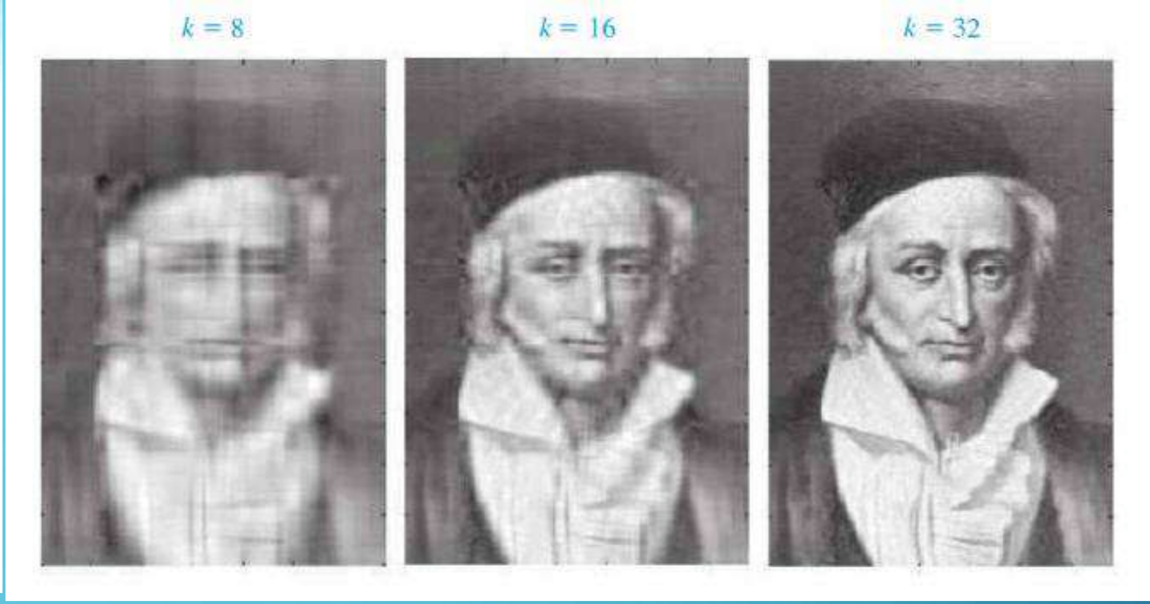
If $A = \sigma_1 R_1 + \sigma_2 R_2 + \cdots + \sigma_r R_r$ is the singular value decomposition of A and

$$A_k = \sigma_1 R_1 + \sigma_2 R_2 + \cdots + \sigma_k R_k$$

Be the $\mathrm{m} \times n$ matrix derived from A with $k \leq r$.

We see the following diagram for various A_k.


Original, k = r = 280
k = 2
k = 4


k = 8
k = 16
k = 32


k = 64
k = 128
k = 256
Bettmann/Corbis

IMAGE COMPRESSION AND LINEAR ALGEBRA

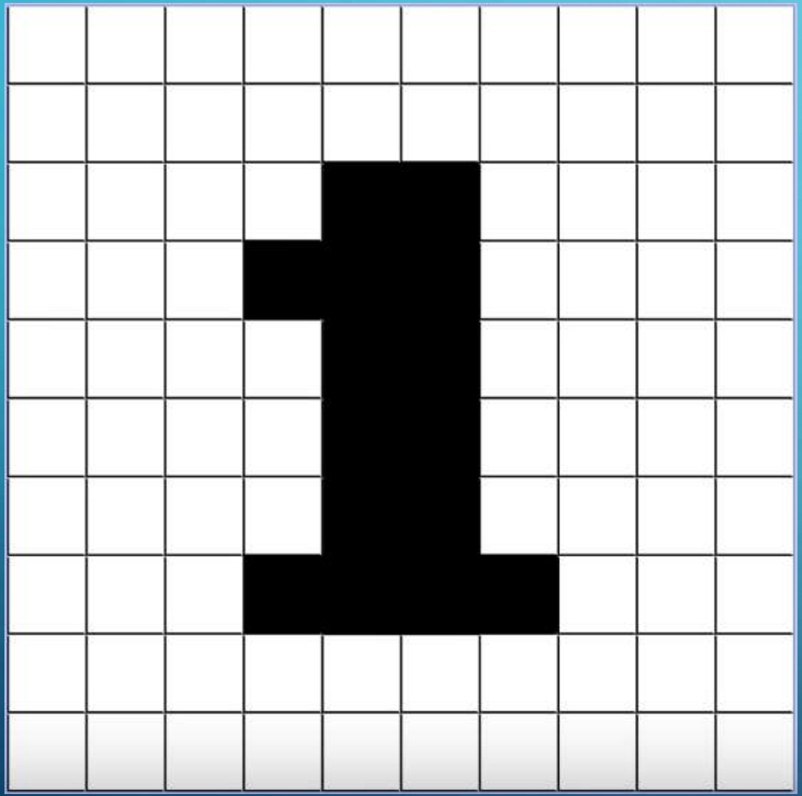
It is clear that for $k = 32$ we almost get the same image. Here A_{32} needs the only

$$32 + (32 \times 340) + (32 \times 260) = 19232$$

Numbers. So, storage is much lesser than original image.

IMAGE PROCESSING

- To make it easier let us look at a black and white picture.



1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1

Black pixel is represented as 0 and white pixel is represented as 1.

IMAGE PROCESSING

- It actually works with a grey scale, means any values between 0 to 1 exists which corresponds to a different shade of grey.

IMAGE PROCESSING

Example of blurring an image

Let's mathematically blur the image of '1'.

1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1

$$* \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix} =$$

Kernel

	1								

Multiply the each element of the red colored 3x3 matrix with the kernel and sum

IMAGE PROCESSING

Example of blurring an image

Multiply the each element of the red colored 3x3 matrix with the kernel and sum

1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	1	0	0	1	1	1	1
1	1	1	0	0	0	0	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1

$$* \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix} =$$

	1	1	.89						

Kernel

IMAGE PROCESSING

Example of blurring an image

1	1	1	1	1	1	1	1	1	1
1	1	1	.89	.78	.78	.89	1	1	1
1	1	.89	.67	.44	.56	.78	1	1	1
1	1	.89	.56	.22	.33	.67	1	1	1
1	1	.89	.56	.22	.33	.67	1	1	1
1	1	1	.67	.33	.33	.67	1	1	1
1	1	.89	.56	.22	.22	.56	.89	1	1
1	1	.89	.67	.44	.44	.67	.89	1	1
1	1	.89	.78	.67	.67	.78	.89	1	1
1	1	1	1	1	1	1	1	1	1

Eventually the matrix turns out as follows

Though this method does not apply for borders, but for our purposes we keep that white.

IMAGE PROCESSING

Example of blurring an image

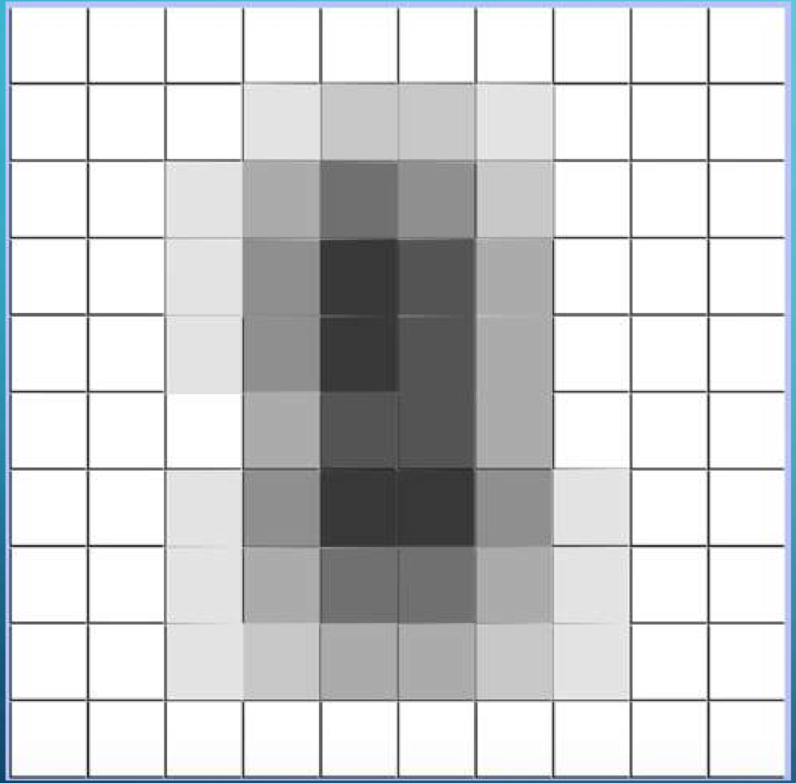
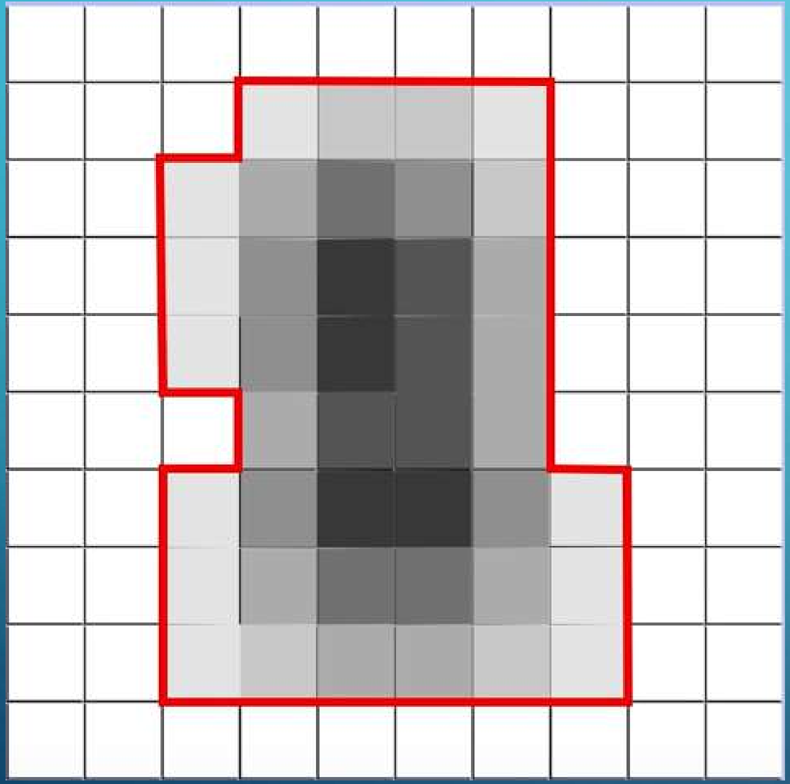


IMAGE PROCESSING

Example of blurring an image



$$* \begin{pmatrix} 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \\ 1/9 & 1/9 & 1/9 \end{pmatrix} =$$

Blur

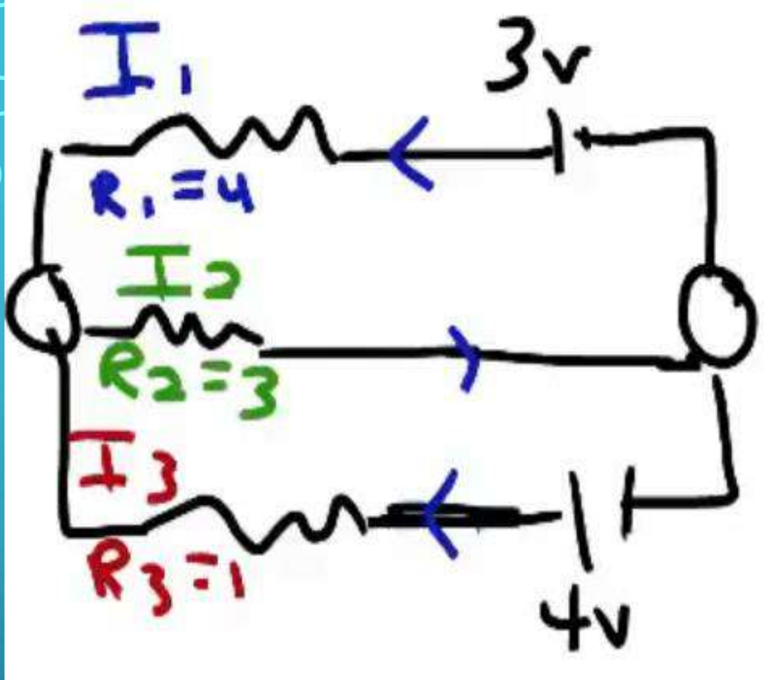
$$* \begin{pmatrix} 0 & -1 & 0 \\ -1 & 5 & -1 \\ 0 & -1 & 0 \end{pmatrix} =$$

Sharpen

$$* \begin{pmatrix} -1 & -1 & -1 \\ -1 & 5 & -1 \\ -1 & -1 & -1 \end{pmatrix} =$$

Edge detection

APPLICATION IN NETWORK MODEL



Find the currents I_1, I_2 and I_3.

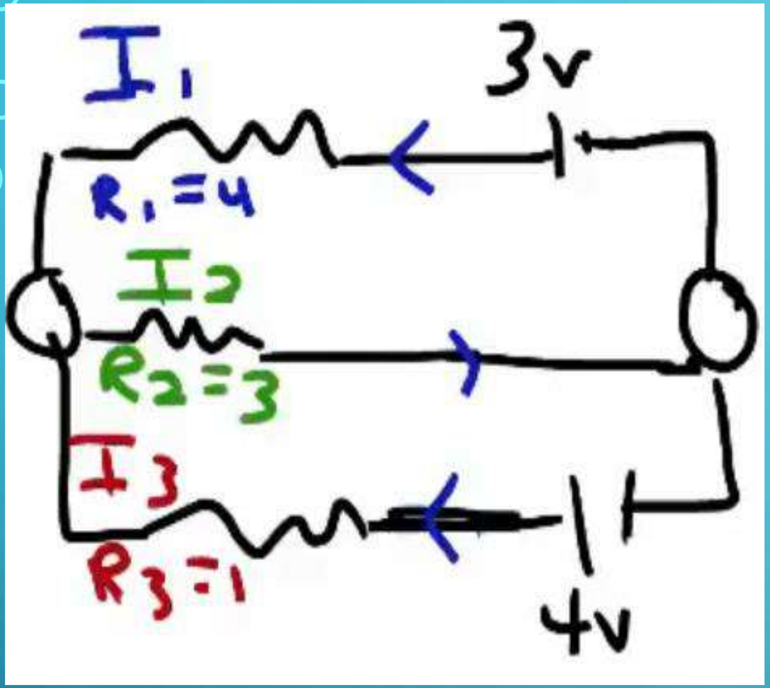
Kirchhoff's Laws:

1. All current that flows into a junction must flow out of it.
2. The sum of the products IR around a closed path equals the total voltage in its path.

Note: I = current and R = resistance

We measure current (I) in amps, resistance (R) in ohms, and IR in volts.

APPLICATION IN NETWORK MODEL

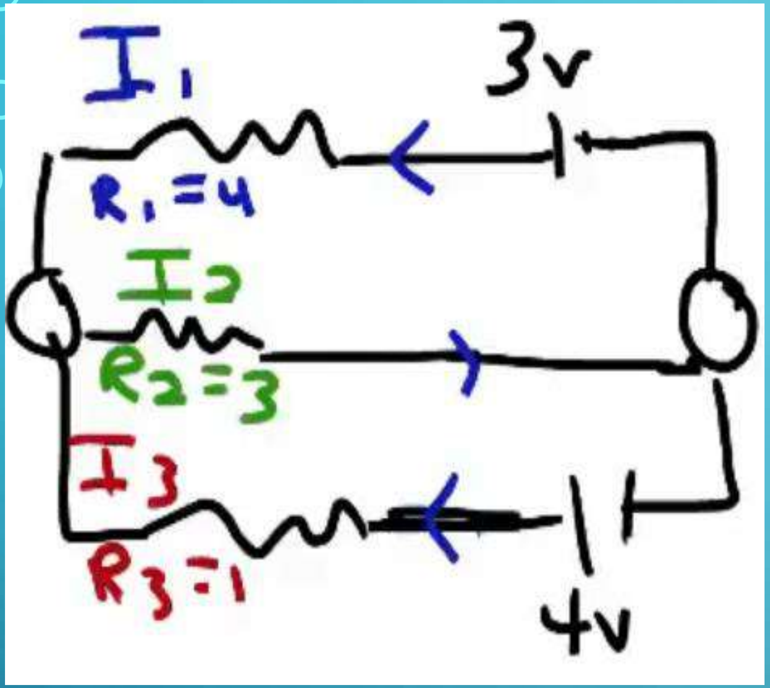


$$I_1 + I_3 = I_2$$

$$4I_1 + 3I_2 = 3$$

$$3I_2 + I_3 = 4$$

APPLICATION IN NETWORK MODEL



Which implies

$$I_1 - I_2 + I_3 = 0$$

$$4I_1 + 3I_2 + 0I_3 = 3$$

$$0I_1 + 3I_2 + I_3 = 4$$

Corresponding matrix is

$$\begin{pmatrix} 1 & -1 & 1 \\ 4 & 3 & 0 \\ 0 & 3 & 1 \end{pmatrix} \begin{pmatrix} I_1 \\ I_2 \\ I_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 3 \\ 4 \end{pmatrix}$$

Which gives $I_1 = 0$ amp, $I_2 = 1$ amp and $I_3 = 1$ amp.

APPLICATION IN CRYPTOGRAPHY:

- Let the message be PREPARE TO NEGOTIATE

P	R	E	P	A	R	E	*	T	O	*	N	E	G	O	T	I	A	T	E
16	18	5	16	1	18	5	27	20	15	27	14	5	7	15	20	9	1	20	5

We assign a number for each letter of the alphabet.

- For simplicity, let us associate each letter with its position in the alphabet: A is 1, B is 2, and so on.
- We assign the number 27 (remember we have only 26 letters in the alphabet) to a space between two words.

The encoding matrix be

$$\begin{bmatrix} -3 & -3 & -4 \\ 0 & 1 & 1 \\ 4 & 3 & 4 \end{bmatrix}$$

APPLICATION IN CRYPTOGRAPHY:

Since we are using a 3 by 3 matrix, we break the enumerated message above into a sequence of 3 by 1 vectors:

$$\begin{bmatrix}16\\18\\5\end{bmatrix} \begin{bmatrix}16\\1\\18\end{bmatrix} \begin{bmatrix}5\\27\\20\end{bmatrix} \begin{bmatrix}15\\27\\14\end{bmatrix} \begin{bmatrix}5\\7\\15\end{bmatrix} \begin{bmatrix}20\\9\\1\end{bmatrix} \begin{bmatrix}20\\5\\27\end{bmatrix}$$

Note that it was necessary to add a space at the end of the message to complete the last vector.

APPLICATION IN CRYPTOGRAPHY:

- Encoding part:

$$\begin{bmatrix} -3 & -3 & -4 \\ 0 & 1 & 1 \\ 4 & 3 & 4 \end{bmatrix} \begin{bmatrix} 16 & 16 & 5 & 15 & 5 & 20 & 20 \\ 18 & 1 & 27 & 27 & 7 & 9 & 5 \\ 5 & 18 & 20 & 14 & 15 & 1 & 27 \end{bmatrix} = \begin{bmatrix} -122 & -123 & -176 & -182 & -96 & -91 & -183 \\ 23 & 19 & 47 & 41 & 22 & 10 & 32 \\ 138 & 139 & 181 & 197 & 101 & 111 & 203 \end{bmatrix}$$

The message is transmitted in the following linear form

$$-122,\ 23,\ 138,\ -123,\ 19,\ 139,\ -176,\ 47,\ 181,\ -182,\ 41,\ 197,\ -96,\ 22,\ 101,\ -91,\ 10,\ 111,\ -183\ \ 32\ \ 203.$$

APPLICATION IN CRYPTOGRAPHY:

- Decoding part:

The transmitted message is

−122, 23, 138, −123, 19, 139, −176, 47, 181, −182, 41, 197, −96, 22, 101, −91, 10, 111, −183 32 203.

To decode the message, the receiver writes this string as a sequence of 3 by 1 column matrices and repeats the technique using the inverse of the encoding matrix.

$$\begin{bmatrix} 1 & 0 & 1 \\ 4 & 4 & 3 \\ -4 & -3 & -3 \end{bmatrix} \begin{bmatrix} -122 & -123 & -176 & -182 & -96 & -91 & -183 \\ 23 & 19 & 47 & 41 & 22 & 10 & 32 \\ 138 & 139 & 181 & 197 & 101 & 111 & 203 \end{bmatrix} = \begin{bmatrix} 16 & 16 & 5 & 15 & 5 & 20 & 20 \\ 18 & 1 & 27 & 27 & 7 & 9 & 5 \\ 5 & 18 & 20 & 14 & 15 & 1 & 27 \end{bmatrix}$$

The columns of this matrix, written in linear form, give the original message:

16	18	5	16	1	18	5	27	20	15	27	14	5	7	15	20	9	1	20	5
P	*R*	*E*	*P*	*A*	*R*	*E*	*	*T*	*O*	*	*N*	*E*	*G*	*O*	*T*	*I*	*A*	*T*	*E*

ASSESSMENTS

- CAT 1 - 30 MARKS -15 MARKS (WEIGHTAGE)
- CAT 2 - 30 MARKS - 15 MARKS (WEIGHTAGE)
- DIGITAL ASSIGNMENT 1 - 10 MARKS
- DIGITAL ASSIGNMENT 2 - 10 MARKS
- QUIZ -1 - 10 MARKS

BEST WISHES