

# **STUDIES ON PERFORMANCES OF DICTIONARY LEARNING ALGORITHMS FOR MEDICAL IMAGE DEBLURRING PROBLEMS**

*A Thesis Submitted in Partial Fulfilment of the Requirements for  
the*

*Degree of Master of Electrical Engineering*

*By*

**MEDHA NAG**

Examination Roll No.: **M4ELE19004**

Registration No. : **140661 of 2017-18**

**JADAVPUR UNIVERSITY**

*Under the guidance of*

**Dr. AMITAVA CHATTERJEE**

**Electrical Engineering Department**

**Faculty of Engineering and Technology**

**JADAVPUR UNIVERSITY**

**KOLKATA- 700032, W.B. INDIA**

**2017-2019**

**JADAVPUR UNIVERSITY**  
**FACULTY OF ENGINEERING AND TECHNOLOGY**  
**ELECTRICAL ENGINEERING DEPARTMENT**  
**KOLKATA- 700032, INDIA**

**CERTIFICATE OF RECOMMENDATION**

I hereby recommend that the thesis titled “**STUDIES ON PERFORMANCES OF DICTIONARY LEARNING ALGORITHMS FOR MEDICAL IMAGE DEBLURRING PROBLEMS**”, submitted by **MEDHA NAG** (Registration No. 140661 of 2017-18), in partial fulfilment of the requirement for the degree of “Master of Electrical Engineering” of Jadavpur University which has been carried out by her under my guidance and supervision. The project, in my opinion, is worthy of its acceptance.

**SUPERVISOR**

---

**Prof. Amitava Chatterjee**  
Professor  
Electrical Engineering Department,  
Jadavpur University

**COUNTERSIGNED**

---

***Prof. Kesab Bhattacharyya***  
Head, Dept. of Electrical Engineering,  
Faculty of Engineering and Technology,  
Jadavpur University

---

***Prof. Chiranjib Bhattacharjee***  
Dean,  
Faculty of Engineering and Technology,  
Jadavpur University

**JADAVPUR UNIVERSITY**

**KOLKATA- 700032, INDIA**

**FACULTY OF ENGINEERING AND TECHNOLOGY**

**ELECTRICAL ENGINEERING DEPARTMENT**

**CERTIFICATE OF APPROVAL\***

*The foregoing thesis is hereby approved as a creditable study of Master of Electrical Engineering and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned does not necessarily endorse or approve any statement made, opinion expressed or conclusion therein but approve this thesis only for the purpose for which it is submitted.*

**Final Examination for Evaluation of the Thesis**

1. \_\_\_\_\_

2. \_\_\_\_\_

(Signature of Examiners)

\*Only in case the thesis is approved

# DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC ETHICS

---

I hereby declare that the thesis entitled “**Studies on performances of dictionary learning algorithms for medical image deblurring problems**” contains literature survey and original research work as part of the course of Master of Engineering studies. All the information in this document have been obtained and presented in accordance with academic rules and ethical conduct.

I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name : MEDHA NAG

Exam Roll no. : M4ELE19004

Registration no. : 140661 of 2017-18

Thesis Name : STUDIES ON PERFORMANCES OF DICTIONARY  
LEARNING ALGORITHMS FOR MEDICAL IMAGE  
DEBLURRING PROBLEMS

Signature with date :

## ACKNOWLEDGEMENT

---

With a great pleasure, I express my deep sense of gratitude to my supervisors, **Prof. Amitava Chatterjee**, Department of Electrical Engineering of Jadavpur University for giving me the invaluable opportunity to work in this exciting field. I am obliged and grateful to him for guidance, suggestions and encouragement throughout the project. I will always remain thankful to him for his patience with me. As a beginner in this field, I learn how to do research work from him.

I would also wish to express my sincere gratitude to **Professor Sugata Munshi, Professor Debangshu Dey, Professor Biswajit Bhattacharyya, Professor Mita Dutta, Professor Gautam Sarkar and Professor Palash Kr. Kundu** Department of Electrical Engineering, Jadavpur University, for their encouragement, advice and motivation during the coursework.

I am also thankful to **Dr. Kesab Bhattacharyya**, Head, Department of Electrical Engineering, Jadavpur University, for providing the necessary facilities for carrying out this research work.

I am taking the opportunity to express my humble indebtedness to **Mrs. Pubali De** research scholar, for her invaluable inputs during this work. I am also thankful to rest of the research scholars of Measurement laboratory for their support throughout the tenure of the research work.

I would like to thank my dear friend **Mr. Banibrata Ghosh, Mr. Arup Kumar Das, Mr. Sanjoy Pandit**, PG scholar, E.E. Department, from whom I received immense support, inexplicable encouragements and assistance. I would like to convey my soulful thankfulness to the rest of the PG scholars of E.E. Department for their moral support during this course work. I am extremely grateful to my parents and my sister for their constant support and motivation, without that I would not have come to this stage. This thesis, a fruit of the combined efforts of my family members, is dedicated to them as a token of love and gratitude.

Above all, it is the wish of the almighty that I have been able to complete this work.

Thank you,

**Medha Nag**

*DEDICATED TO MY  
FAMILY*

# CONTENTS

---

<i>Certificate of Recommendation</i>	ii
<i>Certificate of Approval</i>	iii
<i>Declaration of Originality and Complains of Academic Ethics</i>	iv
<i>Acknowledgements</i>	v
<i>Contents</i>	vii
<i>List of Figures</i>	x
<i>List of Tables</i>	xi

---

<b>CHAPTER</b>	<b>PAGE</b>
<b>1. INTRODUCTION</b>	1
1.1.BACKGROUND	2
1.2.MOTIVATION OF THESIS	2
1.3.CONTRIBUTION OF THESIS	3
1.4.ORGANIZATION OF THESIS	3
<b>2. IMAGE RECOVERY</b>	5
2.1.INTRODUCTION	6
2.2.IMAGE DEBLURRING	7
2.3.METHOD OF IMAGE DEBLURRING	7
2.4.NEED OF DEBLURRING	8

<b>3. SPARSE REPRESENTATION</b>	9
3.1.GENERAL INTRODUCTION	10
3.2.THE SPARSE MODEL	11
3.2.1. RELAXATION METHOD	15
3.2.2. GREEDY METHOD	15
3.3.CONCLUSION	16
<b>4. DICTIONARY LEARNING</b>	17
4.1.THE BASICS	18
4.2.THE KSVD ALGORITHM	20
4.3.ONLINE DICTIONARY LEARNING METHOD	24
4.4.BLOCK PROXIMAL GRADIENT METHOD	27
<b>5. IMAGE DEBLURRING WITH PATCH DICTIONARY METHOD</b>	30
5.1.INTRODUCTION	31
5.2.IMAGE DEBLURRING	33
5.3.THE MODEL	34
5.3.1.CHOICE OF PATCHES	35
5.3.2.ASSUMPTIONS TECHNIQUES	35
5.4.PROCEDURE OF PATCH DICTIONARY METHOD	37
<b>6. EXPERIMENTAL STUDY</b>	39
6.1.INTRODUCTION	40
6.2.IMPLEMENTATION	40
6.3.EXPERIMENTAL TEST	41
6.4.PERFORMANCES ANALYSIS AND COMPARIOSON OF VARIOUS RESULT	41
6.5.INTELLIGENT METHOD	82



6.6.CONCLUSION	84
<b>7. CONCLUSION</b>	<b>85</b>
7.1.CONCLUSION	86
7.2.FUTURE SCOPE	86
<b>REFERENCES</b>	<b>87</b>

# LIST OF FIGURES

<b>Fig.NO.</b>	<b>DESCRIPTION</b>	<b>PAGE</b>
2.1	Model of image degradation	6
3.1	The sparse representation model	12
3.2	Sparsity for different values of p	13
3.3	Flow chart for derive sparsity	14
4.1	Steps of K-SVD dictionary learning method	21
5.1	Image partition by patches	36
6.1-6.48	Variation in PSNR performance for different values of w	42-69

# LIST OF TABLES

TABLE	DESCRIPTION	PAGE
6.1	PSNR Value chart for MRI images – BPG( $\lambda=0.8$ ) (1% noise)-average blur	42
6.2	PSNR Value chart for CT scan images – BPG ( $\lambda=0.8$ ) (1% noise)-average blur	43
6.3	PSNR Value chart for MRI images – BPG ( $\lambda=0.1$ ) (1% noise)-average blur	44
6.4	PSNR Value chart for CT scan images – BPG ( $\lambda=0.1$ ) (1% noise)-average blur	45
6.5	PSNR Value chart for MRI images – KSVD( $r=8$ ) (1% noise)-average blur	46
6.6	PSNR Value chart for CT scan images – KSVD( $r=8$ ) (1% noise)-average blur	47
6.7	PSNR Value chart for MRI images – KSVD( $r=6$ ) (1% noise)-average blur	48
6.8	PSNR Value chart for CT scan images – KSVD( $r=6$ ) (1% noise)-average blur	49
6.9	PSNR Value chart for MRI images – OLM16 (1% noise)-average blur	50
6.10	PSNR Value chart for CT scan images – OLM16 (1% noise)-average blur	51
6.11	PSNR Value chart for MRI images – OLM32 (1% noise)-average blur	52
6.12	PSNR Value chart for CT scan images – OLM32 (1% noise)-average blur	53
6.13	COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (MRI_2) (NOISE RATIO - 1%)-for different dictionaries	54

6.14	COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (CT_3) (NOISE RATIO - 1%)-for different dictionaries	56
6.15	PSNR Value chart of MRI images – BPG ( $\lambda=0.1$ ) (5% noise)-average blur	58
6.16	PSNR Value chart of CT scan images – BPG ( $\lambda=0.1$ ) (5% noise)-average blur	59
6.17	PSNR Value chart of MRI images – BPG ( $\lambda=0.8$ ) (5% noise)-average blur	60
6.18	PSNR Value chart of CT scan images – BPG ( $\lambda=0.8$ ) (5% noise)-average blur	61
6.19	PSNR Value chart of MRI images – KSVD ( $r=8$ ) (5% noise)-average blur	62
6.20	PSNR Value chart of CT scan images – KSVD( $r=8$ ) (5% noise)-average blur	63
6.21	PSNR Value chart of MRI images – KSVD ( $r=6$ ) (5% noise)-average blur	64
6.22	PSNR Value chart of CT scan images – KSVD( $r=6$ ) (5% noise)-average blur	65
6.23	PSNR Value chart of MRI images – OLM16 (5% noise)-average blur	66
6.24	PSNR Value chart of CT scan images – OLM16 (5% noise)-average blur	67
6.25	PSNR Value chart of MRI images – OLM32 (5% noise)-average blur	68
6.26	PSNR Value chart of CT scan images – OLM32 (5% noise)-average blur	69
6.27	COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (MRI_2) (NOISE RATIO - 5%)-for different dictionaries	70
6.28	COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (CT 3) (NOISE RATIO -5%)-for different dictionaries	72

6.29	COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO - 1%)-for different dictionaries	74
6.30	COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -1%)-for different dictionaries	76
6.31	COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO - 5%)-for different dictionaries	78
6.32	COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -5%)-for different dictionaries	80
6.33	CHART FOR BEST PSNR VALUE FOR SPECIFIC WEIGHT (MRI) & (CT)	83

# **CHAPTER-1**

## ***INTRODUCTION***

- BACKGROUND
- MOTIVATION OF THESIS
- CONTRIBUTION OF THESIS
- ORGANIZATION OF THESIS

# CHAPTER-1

## ***1.1. BACKGROUND:***

In image processing digital images have affected by noise during image acquisition and image transmission. An image can degrade by a variety of factors, such as quality of the sensing element and environmental condition during image acquisition. In camera light levels, sensor temperature are major factor to affect the image. Images may corrupt during transmission due to inference in channel. In this work, the ultimate goal image restoration is required.

The image recovery techniques have applied for improving an image in some pre-defined way. Recovery attempts to reconstruct an image that have degraded by using a prior knowledge of the degradation phenomenon. By image recovery, an optimum estimated desired result has proposed which is very close to the ideal image [1].

## ***1.2. MOTIVATION OF THE THESIS:***

Various algorithms have been proposed for image recovery. Among those, a fast patch dictionary method is proposed in the work carried out by Xu and Yin in [3, 4]. To recovery the whole image, divide the image in many non-overlapping patches. Sparse recovery perform on a subset of non-overlapping patches and then the recovery results are averaged to get the final recovery. This method is very effective for speed and quality. The application of this image patch methods are denoising, inpainting, super resolution, decomposition and deblurring.

In dictionary learning, block proximal gradient method accelerate computation and reduce per iteration complexity. So image inpainting, compression, denoising and deblurring are effective for image processing by combining whole image recovery process and dictionary learning method [3, 4].

Among all these applications, an extension study of deblurring of a blur and noisy image is presented in this work.

The focus in this thesis work is medical image recovery. . Medical imaging is a challenging area image processing. The aim of the thesis is to deblurr the Computed tomography (CT) scan and Magnetic resonance imaging (MRI) image of brain.

This thesis describes the methods to improve the quality of blurred and noisy MRI and CT scan images due to average and motion artifact. The clinical MRI and CT data has normally corrupted by noises and blurred during the measurement process. The main objective of this work is to restore original images from blurred images due to defocused optical system and motion artifact, which is a challenging problem in medical imaging. Among several techniques to restore the original image corrupted by various noise and blurred due to motion artifact, this method analyze sparse recovery with a specific dictionary to fetch the originality of image. The comparisons of the techniques done by using different dictionaries with different weight based on Peak signal to noise ratio (PSNR).

### ***1.3. CONTRIBUTION OF THE THESIS:***

This thesis makes a detail systematic study of medical image deblurring by dictionary learning algorithms. The main inspiration of the study carried out is derived from the work carried out by Xu and Yin in [3, 4]. These works by Xu and Yin ably showed how patch based dictionary learning algorithms can be successfully employed for image recovery problem, considering general images. This thesis work has concentrated on a specific category of image recovery problems i.e. image deblurring problems and the study has concentrated on deblurring of two popular categories of medical images i.e. MRI and CT images. Inspired from the works carried out by Xu and Yin, this thesis has extensively studied how the three dictionary learning algorithms considered by them i.e. KSVD algorithm, Online Dictionary Learning algorithm and Block Proximal Gradient Method will perform in deblurring medical images specifically MRI and CT images. The MRI and CT images considered in this thesis work are considered from benchmark medical image database repositories available in web in [9] and [10]. The thesis has also extensively studied how the performances of these three dictionary algorithms will vary with different choices of an important free parameter i.e. the entries in a weight vector  $\mathbf{w}$  and then finally has proposed an intelligent method of a suitable choice of the weight vector  $\mathbf{w}$  for different deblurring of different medical images that have been considered in this thesis.

### ***1.4. ORGANIZATION OF THE THESIS:***

The entire thesis work is documented in seven different chapters which includes as follows

**Chapter 1:** comprises the background, motivation of the thesis and contribution of thesis work.



**Chapter 2:** studies of image recovery, image deblurring methods and also describes about need.

**Chapter 3:** provides the basic idea about sparse representation in image processing.

**Chapter 4:** describes the dictionary learning algorithm and propose the technique of three different dictionaries -KSVD, OLM and BPG.

**Chapter 5:** elaborates the patch dictionary method for image deblurring. Also describes the model and algorithm of the process.

**Chapter 6:** provides the comparison result based on PSNR value for different noise ratio, weight variable, blurring method of medical image.

**Chapter 7:** provides the conclusions and discuss about the scope for future work briefly.

# **CHAPTER-2**

## ***IMAGE RECOVERY***

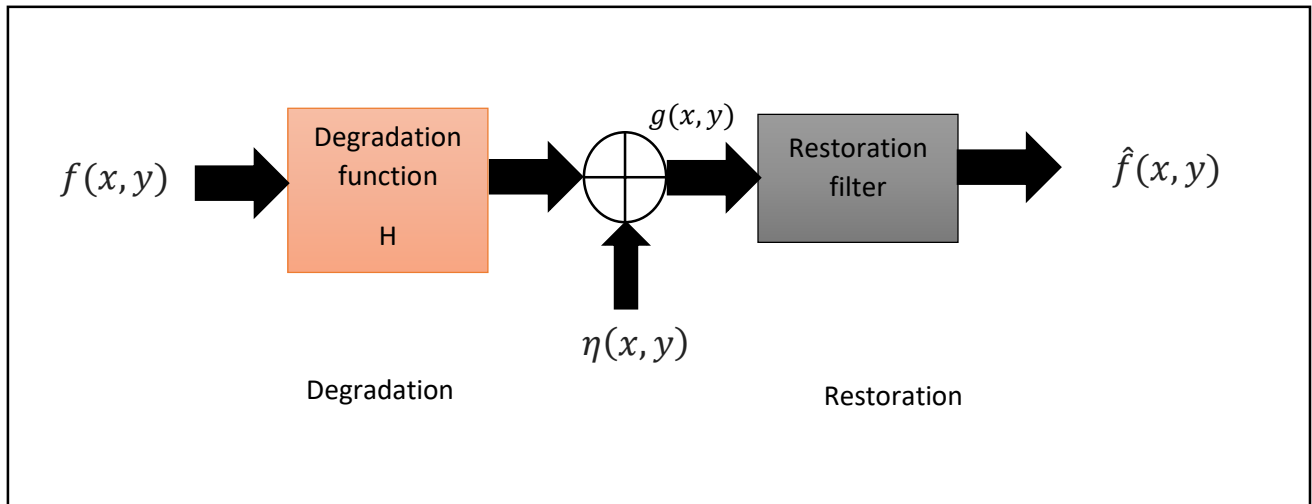
- INTRODUCTION
- IMAGE DEBLURRING
- METHOD OF IMAGE DEBLURING
- NEED OF DEBLURING

## CHAPTER-2

### 2.1. INTRODUCTION:

Image recovery techniques are mainly restoration process. Almost all images suffer from some noise effects. Those are caused by the sensor and the surrounding electronics. In any image, corruption may come in many forms such as motion blur, noise and camera focusing problem. Therefore, we need to remove the noise from the images [5].

Basically, image restoration is an inverse technique, which recover the real image from the degraded one. The objective of image restoration techniques is to reduce noise and recover resolution loss. As fig 2.1 shows, the degradation process is implemented [1]. Here  $f(x, y)$  is an input image, and  $g(x, y)$  is the degraded image, which affected by the degradation function  $H$ , and some noise  $\eta(x, y)$ . The objective of the restoration process is to obtain an estimate  $\hat{f}(x, y)$ . The approach is to estimate to be as close as possible to the original input image.



*Fig.2.1. Model of image degradation [1]*

There are some advanced techniques such as image enhancement, deblurring, denoise, and super resolution have been developed to improve image quality post-digitization. This thesis makes a detail systematic study of deblurring process.

## ***2.2. Image Deblurring:***

Deblurring is one of the significant steps in the image analysis. Generally, at the time of image acquisition process, images may corrupted or degraded due to various factors such as blurring, loss of data due to sampling process, environment and sources of noise which results in image degradation. The various types of blur are Gaussian blur, motion blur etc. [6]. So different techniques of deblurring have been proposed to deblur the degraded image.

## ***2.3. Methods of Deblurring:***

Image deblurring can broadly classified into two categories - blind deconvolution and non-blind deconvolution. There are also some other methods.

- **Blind Deconvolution:** Blind deconvolution attempts to recover the original sharp photograph from a blurred image where the cause of blur is usually modeled as very simple functions such as constant velocity motion or linear harmonic motion. For the motion deblurring problem this method cannot be effectively applied in case of complex PSF since the restored images are degraded by strong deconvolution artifacts and so exact deconvolution is impossible. The restored image may not be perfect as the original one. The obtained solution is not be unique [25].
- **Non-blind deconvolution:** Non-blind deconvolution is also known as the classical linear image restoration problem. In non-blind deconvolution, the blur kernel is known or given to the user. There are two main difficulty. These are presence of noise in the blurred image, and the ripple-like ringing effects around the edges [27].
- **Weiner filter:** Weiner filter acts as an all pass filter. The power spectrum is equal to the variance of the noise for white additive noise. Weiner filter minimizes the mean square error has the capability of handling both the degradation and random noise [25].
- **Richardson-Lucy algorithm:** Richardson-Lucy algorithm works only with images blurred by a valid point spread function. This method conserves energy both globally and locally in all iteration. . It is an iterative method to restore blurred image in this process [6].

- Inverse filter: Inverse filter is a high-pass filter. Inversefiltering is very accurate and easy way to recover animage provided the user knows about blurringfilter. The advantage is that, it just requires only the blur PSF as a priori knowledge which in-turn produces perfect reconstruction in the absence of noise [1].

#### ***2.4. Need for Image Deblurring:***

Degradation of images for various noise is one of the major problems in image processing. Blurring is an unwanted reduction, which degrades the image quality and it is difficult to avoid. Image deblurring has various applications in different fields like medical imaging, forensic science, and astronomy. Various methods have been developed by various researchers for image deblurring. Image deblurring is a challenging issue in image processing. The work is on the medical images which is blur for different issues. Here proposed an extended study of sparse representation method based on dictionary learning process to overcome this issues carried out by Xu and Yin in [3, 4].

# CHAPTER-3

## *SPARSE REPRESENTATION*

- GENERAL INTRODUCTION
- THE SPARSE MODEL

## CHAPTER-3

### 3.1. GENERAL INTRODUCTION:

Sparse Representation is a new and highly effective data model, along with the required powerful and flexible tools for using it in image processing. Almost every task in signal and image processing relies on this model. Sparse representation is relying on mathematical concepts studied in approximation theory, and tools taken from linear algebra and optimization. Moreover, beyond these, Sparse representation borrows ideas from machine learning, in the context of adapting the model to a given data source. The product of all these foundations is a model that has been shown to effectively treat numerous applications, leading in all of these to state-of-the-art results. This field of sparse representations gathered its main achievements during the past two decades. Sparse Modeling is used for many, many other areas in image processing, but here the work is for image denoising. In denoising problem sparse representation technique is very effective. [20]

$$f(\mathbf{X}) = 1/2\|\mathbf{X} - \mathbf{Y}\|_2^2 + G(\mathbf{X}) \quad [20] \quad (3.1)$$

Where,  $\mathbf{Y}$ =noisy image

$\mathbf{X}$  =recovered image

$G(\mathbf{X})$ = prior

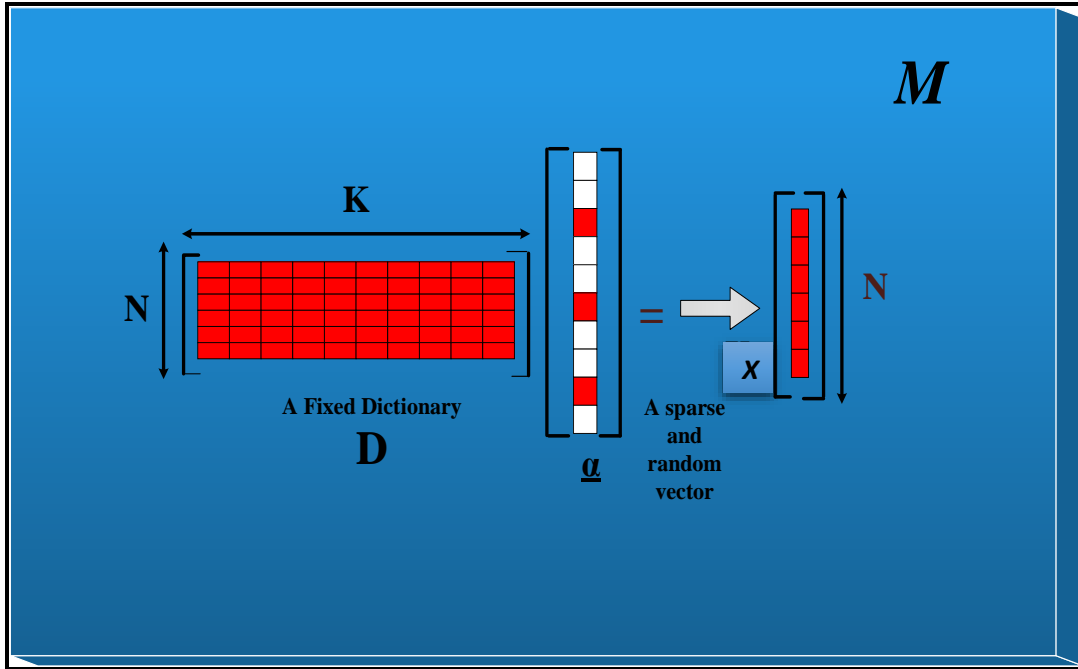
To recover a clean image, that is  $\mathbf{x}$  and do not want the recover image to be too far away from the noisy image. And here this is means square error, between the image that observe and the image that we recover. The motivation is to minimize the means square error. Here another term  $G(\underline{x})$  has multiple names as prior or regularization term.

Here sparse representation is use as a good prior.

### 3.2. *THE SPARSE MODEL:*

Sparse coding is a modelling data vector, which is sparse linear combination of basis element, this sparse coding hugely used in machine learning, signal processing, statistic etc. This is a very simple diagram (3.1) to present the sparse model. The signal  $\mathbf{X}$  that has  $N \times 1$  dimensions. If we consider  $8 \times 8$  blocks then  $N$  is equal to 64. We take an  $8 \times 8$  block or image patch and we take one row after another. We concatenate them and we get  $N = 64$ . One of the main components of Sparse Modelling is the Dictionary. The Dictionary is a matrix, which is  $N \times K$ .  $N$  is the dimension of the signal we are looking to model.  $K$  is the size of the dictionary having  $K$  columns. Every column is called an atom. It is kind of an image, kind of a patch. Every column is 64-dimensional. Also we have  $K$  such columns. Very often,  $K$  is larger than  $N$ . If  $K$  larger than  $N$ , then we say that the dictionary is over complete. Now, if  $K$  is equal to  $N$ , then the dictionary is complete and If  $K$  is less than  $N$ , we say that a dictionary is under-complete. The second element of sparse modeling is the vector alpha. And the vector, alpha, that has to have dimensions,  $K$ , and we multiply the matrix,  $\mathbf{D}$ , the dictionary, by the vector, alpha, and we produce the signal that we're trying to model. This vector alpha, is the number of non-zero entries of the vector is very small, represent those by this red dot. Here  $L$  the non-zero entries and we multiply this matrix, the Dictionary  $\mathbf{D} \times \alpha$ . We have a linear combination according to the coefficients here of those  $L$  atoms. And because these are only a few non-zero coefficients, this vector is very sparse [20].





*Fig.3.1. The sparse representation model [20]*

This is the concept of sparse modeling.

Now if there is a dictionary of size  $K$ , meaning  $K$  atoms, and have to pick  $L$  means having  $L$  choose out of  $K$  possibilities. So if  $L$  is three, can pick any three out of  $K$ . Once picking them, each one of those selections define a subspace, a very low dimensional subspace. Then the signal, it is just a linear combination of the corresponding atoms and that is a linear subspace. But here  $L$  out of  $K$ , a lot of them. So the model is extremely, extremely rich. [21].

Our signal  $\mathbf{X} = \mathbf{D}\alpha$ ,

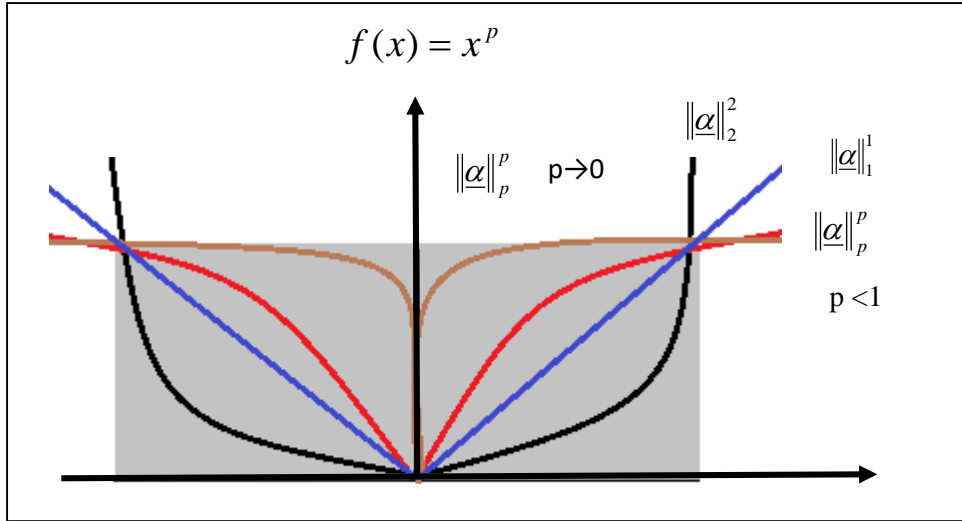
Where  $\alpha$  is a sparse vector.

Now to measure Sparsity, have to calculate the LP norm, which is for a given  $p$ .

$$\|\alpha\|_p^p = \sum_{j=1}^k |\alpha_j|^p \quad (3.2)$$

Here  $\alpha$  is nothing else than a  $k$  dimensional vector. And here with this formula is that taking every entry of  $\alpha$ , and calculate absolute value, and elevate to the  $p$  power, and add all of them, and that's the LP norm of the vector.

Now calculate this for different values of  $p$  is shown in Fig 3.2 [20].



*Fig.3.2. Sparsity for different values of p [20].*

If p is equal to two, do not really get penalization with an equal amount, every time one of the entries of alpha is non-zero. Then the penalization increases quadratically with actually demanding to lose the entry. When p is equal to one the penalization now is proportional, linearly proportional to the magnitude of the entry, is not quadratic.

If now, p becomes less than one, then occurs an equal amount of penalization. If further, decrease p less than one, getting close to zero, get equal penalty for everybody that is non-zero. So this is the way going to measure sparsity with p= 0 with this function.

As  $p \rightarrow 0$  have to count of the non zeros in the vector  $\|\alpha\|_0^0$

$\|\alpha\|_0^0$  is called a  $\iota_0$  pseudo norm, going to count the number of non-zero elements in the vector alpha.

So our signal  $\mathbf{X} = \mathbf{D}\alpha$ , where  $\|\alpha\|_0^0 \leq L$ .

Now to solve the sparse vector  $\underline{\alpha}$  :-

$$\alpha = \underbrace{\underset{\alpha}{\operatorname{argmin}}} 1/2 \|\mathbf{D}\alpha - \mathbf{Y}\|_2^2 \text{ s.t. } \|\alpha\|_0^0 \leq L, \quad (3.3)$$

Where  $\hat{\mathbf{X}} = \mathbf{D}\hat{\alpha}$  and  $\hat{\alpha}$  is optimal.

Here is not allowed to pick any alpha, only allowed to pick alpha that has at most L non-zeros, which means that have to be approximating Y by, at most L columns, the linear combination of at most L columns of **D**.

Now minimize the error under the constraint that you are not allowed to pick more than L non-zero entries in alpha.

This problem can solve by the way-

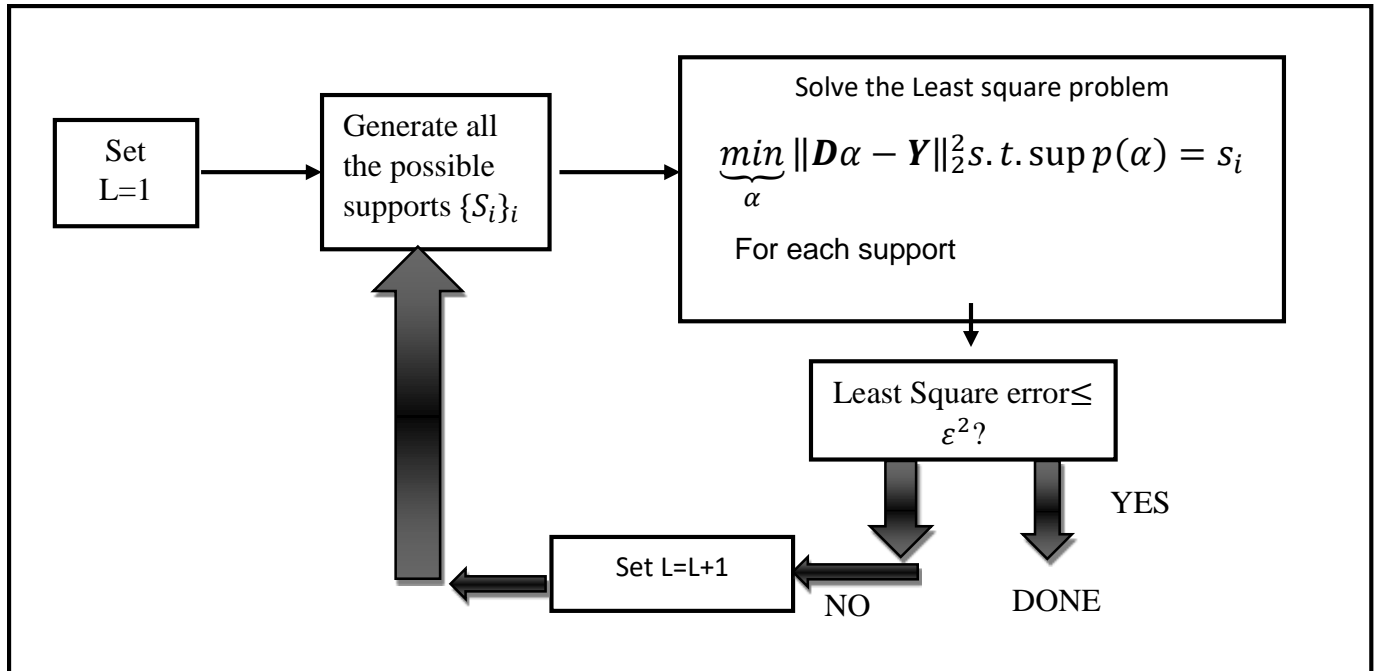
$$\underbrace{\min}_{\alpha} \|\alpha\|_0^0 \text{ s. t. } \|D\alpha - Y\|_2^2 \leq \varepsilon^2 \quad (3.4)$$

Here the error should not more than the given error and find the sparse possible vector that achieves this error.

So now have to choose the dictionary and optimize the alpha.

This problem is called NP Hard problem [20].

Steps for solving this problem:-



*Fig.3.3. Flow chart for derive sparsity [20]*

This is the flow chart for solving the NP Hard problem in Fig 3.3. But if the K is big, then have to calculate a very big large number. Assume that each one of the computations takes about one Nano

second. So, trying all possible supports of size need a long, long time to solve this problem. That is why the problem is very hard. It is impossible to solve as it is.

There are two way to solve this problem:-

1. Relaxation Method

2. Greedy Method

### **Relaxation Method**

Relaxation methods are also sometimes called basis pursuit methods [20, 22]. Relaxation means relax some conditions in such a way that is almost the same, but can solve it.

Instead of solving

$$\underbrace{\min}_{\alpha} \|\alpha\|_0 \text{ s. t. } \|D\alpha - Y\|_2^2 \leq \varepsilon^2 \quad (3.5)$$

Now in Relaxation method have to solve

$$\underbrace{\min}_{\alpha} \|\alpha\|_1 \text{ s. t. } \|D\alpha - Y\|_2^2 \leq \varepsilon^2 \quad (3.6)$$

Here replacing the zero norm or pseudo norm by one norm. So now not only just counting the number of knowns zero coefficients, counting with their magnitude.

In this L0 norm, the penalty increase with the magnitude. In a certain condition of D and L the level of sparsity of this two is equivalent. Now the NP-Hard problem can be solvable because this is a convex problem and in reasonable time. So without losing anything the solving procedure is possible.

### **Greedy Method**

The greedy approach is often called the matching pursuit [20, 23] and the idea is very easy. The matching pursuit [21] is a standard technique in the statistics community and it have brought into signal processing about twenty years ago. The basic idea is finding the most important atom in the dictionary .The most important means for which atom  $\|D\alpha - Y\|_2^2$  is minimum.

### Steps

1. Find the one atom, which is best
2. Given the previously found atoms, find the next one to best fit the residual.
3. The algorithm stops when the error  $\left\| \underline{D\alpha - y} \right\|_2^2$  is below the destination threshold.

### 3.3 CONCLUSION:

Sparse representation are obtained on a basis, which takes advantage of some form of regularity of the input signals, creating many small-amplitude coefficients. It is actually a prior used in regularization. Sparse representations such as wavelets or curvelets, which have been very successful for 2-D image processing. Sparse representation is a very effective way in image processing area now a days. In recovery of degraded image the sparse coding is proposed here.

# **CHAPTER-4**

## ***DICTIONARY LEARNING***

- THE BASICS
- THE KSVD ALGORITHM
- ONLINE DICTIONARY LEARNING METHOD
- BLOCK PROXIMAL GRADIENT METHOD

# CHAPTER-4

## **4.1. THE BASICS:**

In this chapter dictionary learning approach is explained in detail which is used with sparse representation, that can be suitably utilized for deblurring the noisy blur MRI and CT scan images. The objective of a dictionary learning algorithm is to learn the dictionary in such a way that can get best suitable representation of each member within that dictionary, strictly maintaining the sparsity constraint.

The dictionary should be chosen such that it sparsifies the representation. There are basically two way to do that-

- Off the shelf-dictionaries –

To choose dictionary from a known set of transform (wavelet, curve let etc.). They are not adopted to the signal, which means they are universal. But they are not the best possible [18].

- Learned dictionaries –

Approach learning from a larger training set, instead of using predefined basis, such as wavelet or Fourier basis or uses the larger training data set itself. In this case, using the entire set of the input training signals as a complete dictionary can give better performance, but it have large computational burden. To overcome this problem a learned dictionary from a large training database and smaller in size than that, can suitably maintain sparsity constraints, but also have desired performance accuracy [18].

### **How to learn a dictionary:**

The dictionary learning problem starts with the accumulation of signals to train on. Suppose we have access to  $N$  such examples, each of these signals is expected to have a  $k_0$  sparse representation with respect to the unknown dictionary. Also, assume that the signals are not perfect, that has deviation.

N signals have been generated from sparse representation with an unknown but fixed dictionary  $\mathbf{D}$  of known size  $n \times m$ :

$$\|\alpha_j\| = k_0, \quad \mathbf{Y}_j = \mathbf{D}\alpha_j + \varepsilon \quad (4.1)$$

These signal and several parameter will be used in this learning process, seeking the best fitted dictionary. The parameter  $k_0$ = number of non zeros in the representation,  $\varepsilon$ = the representation error,  $m$ = the number of atoms  $\mathbf{D}$  has,  $\alpha_j$ = signal representation.

Now the goal is to get both get  $\mathbf{D}$  and the representation for matching. To solve the task there are two popular option. The first one is searching smallest average sparsity, when epsilon has a limitation-

$$\min_{\mathbf{D}, \{\alpha_j\}} \sum_{j=1}^N \|\alpha_j\|_0 \text{ s.t. } \forall j, \|\mathbf{Y}_j - \mathbf{D}\alpha_j\|_2 \leq \varepsilon \quad (4.2)$$

The second one is minimizing the average representation error when sparsity has a limitation:

$$\min_{\mathbf{D}, \{\alpha_j\}} \sum_{j=1}^N \|\mathbf{Y}_j - \mathbf{D}\alpha_j\|_2 \text{ s.t. } \forall j, \|\alpha_j\|_0 \leq k_0 \quad (4.3)$$

The choice between two options depends on the availability of  $k_0$  or epsilon.

Now for the second option there is no role of the order of atoms of  $\mathbf{D}$ . So any permuted version of same dictionary can estimate same. Also there is a scale ambiguity between  $\mathbf{D}$  and representation. To follow the L2 norm of the columns of  $\mathbf{D}$  is a remedy of second flaw [18].

The above core idea may be proposed in different ways through several variation of dictionary learning. In this work for purpose three learned different dictionary learning method used for deblurring the noisy image.

- The KSVD algorithm
- Online dictionary learning method
- Block proximal gradient method

There are some brief explanation about these three dictionaries.



#### 4.2. THE KSVD ALGORITHM:

Now the present approach to represent the KSVD a dictionary learning algorithm. KSVD is an over complete dictionary and kind of an extension of K means for clustering. The goal is to find the dictionary  $D$ , which provides sparse representations for the training signals.

The focus of K-SVD dictionary learning algorithm is to minimize the following objective function [3, 14]:

$$\arg \min_{D, X} \|Y - DX\|_F^2 \text{ subject to } \forall i, \|x_i\|_0 \leq T_0 \quad (1) \quad (4.4)$$

Where  $T_0$  is called sparsity prior. The number of nonzero elements in  $x_i$  each *should* be less than  $T_0$ . Basic K-SVD algorithm proposed a reconstructive dictionary where the dictionary is composed of  $K$  prototypes atoms each of dimension  $n$ . We assume that the input signal matrix  $Y$  comprises  $N$  input signals, each of dimension  $n$ .

$$Y = [y_1, y_2, \dots, y_N] \in R^{n \times N} \quad (4.5)$$

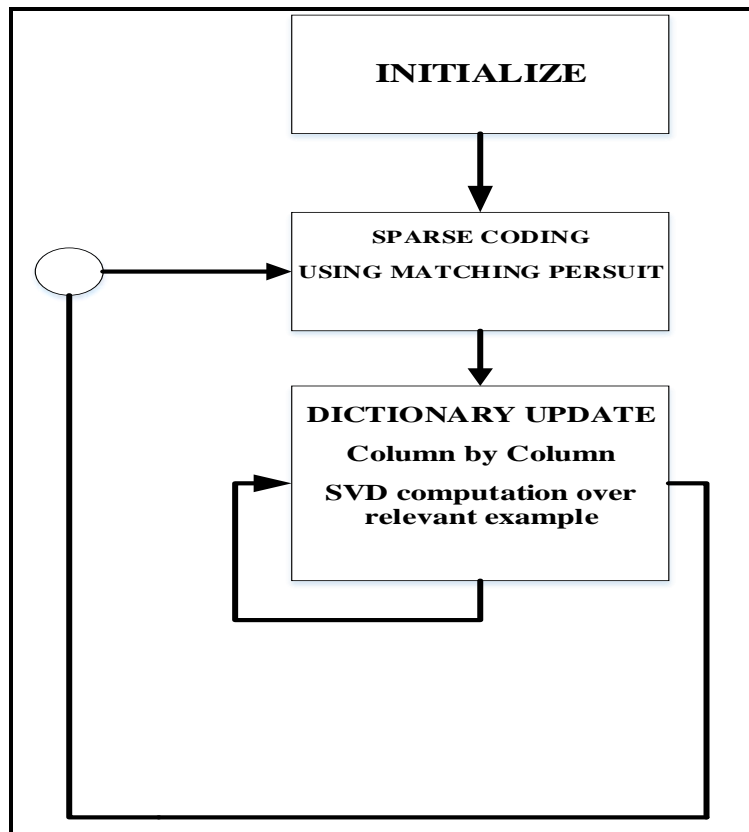
$$\text{And dictionary } D = [d_1, d_2, \dots, d_K] \in R^{n \times K} \quad (4.6)$$

Where  $K \gg n$  that makes that the dictionary is, over-complete. Also, it is chosen that  $N \gg K$ , so that the dictionary, playing the role of a codebook, comprises  $K$  code words, that can be efficiently represent the original training database. The sparse coding of input signal  $Y$  produces

$$X = [x_1, x_2, \dots, x_N] \in R^{K \times N} \quad (4.7)$$

In [20] from the fig (4.1) to learn a dictionary, first initialize the dictionary, anyway. The way of initializing is just to pick randomly, some of the data points. So picking to learn  $K$  atoms. There are  $P$  signals, and  $P$  is larger than  $K$ . So just randomly picking here,  $K$  of the signals and put them as dictionary. That is an initialization. Now, the next step is do Sparse Coding with that dictionary. Now basically have to go every signal and have to solve as Sparse Coding problem. First go to the code for the first signal, then code for the second signal. Thus, the sparse coding will be processed. Now, have to go all around and process sparse coding for each one of these. The next part is activate the dictionary. Now have to go in the other direction, for doing this Sparse Coding study

for every signal. In this dictionary, update one item at a time is possible. So update one atom at a time. Now after doing updated, going to let the signals, to encode again and reiterate. The process will be going on until converging a prefixed number of iteration depending on our computational capabilities.



*Fig.4.1.Steps of K-SVD dictionary learning method [20]*

Now here the each of one these block explained below of fig (4.1).

In sparse coding representation, the best possible way to determine sparse coefficient matrix  $\mathbf{X}$  to keep the dictionary  $\mathbf{D}$  static. Here, in above the objective function in (4.4) can present in the updated form, given as:

$$\|Y - DX\|_F^2 = \sum_{i=1}^N \|\mathbf{y}_i - D\mathbf{x}_i\|_2^2 \quad (4.8)$$

Orthogonal matching pursuit (OMP) [15] algorithm is one of the popular choice for the sparse coding stage, although any other pursuit algorithms can also be effectively employed for this

purpose. In the next dictionary update stage, dictionary  $\mathbf{D}$  can restructure iteratively by updating one atom at a time, keeping  $\mathbf{X}$  fixed. These two are the basic stages, used in an iterative fashion, for minimizing the objective function in K-SVD algorithm, given in (4.4). At any point of time, in the dictionary update stage,  $\mathbf{x}_T^k$  denoted as a specific atom  $\mathbf{d}_k$  of the dictionary and its corresponding coefficients in  $\mathbf{X}$  in its  $k$ th row.

Then the objective function in (4.4) can be written as [17]:

$$\|\mathbf{Y} - \mathbf{D}\mathbf{X}\|_F^2 = \|\mathbf{E}_k - \mathbf{d}_k \mathbf{x}_T^k\|_F^2 \quad (4.9)$$

Where  $\mathbf{E}_k$  presents the error, involved for all  $N$  samples, after removal of the  $k$ th atom of  $\mathbf{d}_k$ . To determine  $\mathbf{d}_k$  and corresponding  $\mathbf{x}_T^k$  by approximating  $\mathbf{E}_k$  singular value decomposition (SVD) algorithm is used. They will not serve the purpose, as the solution, that produced will not satisfy the sparsity constraints in the first place. So, respecting to the sparsity constraints, for obtaining a solution a set of indices  $\mathbf{w}_k$  is formed, which denotes the positions, where the array elements of  $\mathbf{x}_T^k$  are nonzero, given as:

$$\mathbf{w}_k = \{(i | 1 \leq i \leq N, \mathbf{x}_T^k(i) \neq 0)\} \quad (4.10)$$

Where,  $\mathbf{x}_T^k$  is the  $k$ th row in  $\mathbf{X}$ .

To form a matrix  $\Omega_k$ , this array  $\mathbf{w}_k$  is utilized, which may be utilized to build a restricted array  $\mathbf{x}_R^k$  from  $\mathbf{x}_T^k$  utilizing the matrix operation  $\mathbf{x}_R^k = \mathbf{x}_T^k \Omega_k$ . From  $\mathbf{x}_T^k$ , this row vector  $\mathbf{x}_R^k$  is essentially formed by removing zero entries from  $\mathbf{x}_T^k$ . This ensures that process, for subsequent operations, any adaptation carried out for  $\mathbf{x}_R^k$  array means have to essentially adapt the nonzero entries in  $\mathbf{x}_T^k$  and all zero entries in  $\mathbf{x}_T^k$  remain unchanged. That is the sparsity constraints are fully respected here. This means respecting sparsity constraints, that the objective function given in (4.10) can now be transform to the equivalent form, given as:

$$\|\mathbf{E}_k \Omega_k - \mathbf{d}_k \mathbf{x}_T^k \Omega_k\|_F^2 = \|\mathbf{E}_k^R - \mathbf{d}_k \mathbf{x}_R^k\|_F^2 \quad (4.11)$$

Now, here the minimization of (4.11) can be directly obtained from SVD, which decomposes the restricted matrix  $\mathbf{E}_k^R$  as  $\mathbf{E}_k^R = \mathbf{U}\Delta\mathbf{V}^T$ . Consequently, the  $k$ th column of the dictionary is updated by choosing the first column of, that is as  $\mathbf{U}(:,1)$  as the solution  $\widetilde{\mathbf{d}}_k$ . The corresponding coefficient vector  $\mathbf{x}_R^k$  is updated, as  $\mathbf{x}_R^k = \Delta(1,1) * \mathbf{V}(:,1)$ , where  $\mathbf{V}(:,1)$  is the first column of  $\mathbf{V}$ . This procedure is followed to update each atom of the dictionary (along with its corresponding  $\mathbf{x}_T^k$ ), one at a time, until all atoms will updated. The complete procedure of the KSVD algorithm employed in this work, present in algorithm 1[17].

---

**Algorithm 1. KSVD Algorithm**

---

**Step 1:**

- **Initialization of Dictionary:**  $\mathbf{D}^0 \in R^{n \times k}$  with input signals and the columns of using  $l_2$  norm  $\mathbf{D}^0$  are normalized.
- **Input:**  $\mathbf{Y} \in R^{n \times k}, T_0, \mathbf{D}^0 \in R^{n \times k}$
- **Output:**  $\mathbf{D} \in R^{n \times k}, \mathbf{X} \in R^{n \times k}$
- **Set**  $J = 1$
- **Repeat** until convergence (stopping Criterion).

**Step 2:**

- **Sparse coding stage:**  $i = 1, \dots, N$

$$\min_{\mathbf{x}_i} \{ \|\mathbf{y}_i - \mathbf{D}\mathbf{x}_i\|_2^2 \} \text{ subject to } \|\mathbf{x}_i\|_0 \leq T_0$$

**Step 3:**

- **Dictionary update stage:**

For each Column  $k = 1, 2, \dots, K$  in  $\mathbf{D}^{J-1}$  Update it by

- Determine indices  $w_k$  using (7)
- Determine  $\mathbf{E}_k = \mathbf{Y} - \sum_{j \neq k} d_j \mathbf{x}_T^j$
- Restrict  $\mathbf{E}_k$  to obtain  $\mathbf{E}_k^R$  by selecting only the columns corresponding to  $w_k$ .

– SVD decomposition is applied as  $E_k^R = U\Delta V^T$

– Update  $\widetilde{\mathbf{d}}_k = \mathbf{U}(:,1)$  and  $\mathbf{x}_R^k$  is updated as

$$\mathbf{x}_R^k = \Delta(1,1) * \mathbf{V}(:,1)$$

• Set  $\mathbf{J}=\mathbf{J}+1$

---

In the next chapters, the result will be present using this K-SVD dictionary learning method.

### 4.3. ONLINE DICTIONARY LEARNING METHOD:

For requiring a learned dictionary, in [15] another popular online dictionary learning (OLM) method proposed. OLM allows more flexibility to adapt the representation to the data. While learning this dictionary, it has proven to be critical to achieve or improve the results, effectively solving the corresponding optimization problem, which is a significant computational challenge. It involves particularly in the context of the largescale datasets in image processing tasks, which can include millions of training samples.

Let, a signal  $\mathbf{x}$  in  $\mathbb{R}^m$  and a sparse approximation over a dictionary  $\mathbf{D}$  in  $\mathbb{R}^{m \times k}$  where  $k$  is columns considered as atoms, when a linear combination can be found in few atoms from  $\mathbf{D}$  that is close to the signal  $\mathbf{x}$ . By experiment, we found that modelling a signal using sparse coding is very efficient in many signal-processing applications. Usually for images, previously defined dictionaries based on different kinds of wavelet. However, this dictionary learning shows better results than using off-the shelf bases method. However, some of the learned dictionary elements may sometimes “look like” wavelets (or Gabor filters), they are tuned to the input images or signals, leading to much better results in practice.

OLM, the online approach attempts to solve [15]

$$\min_{\mathbf{D}, \mathbf{Y}} \frac{1}{2} \|\mathbf{D}\mathbf{Y} - \mathbf{X}\|_F^2 + \lambda \|\mathbf{Y}\|_1, s. t. \|\mathbf{d}_i\|_2 \leq 1, i = 1, \dots, K \quad (4.12)$$

Where  $\|\mathbf{Y}\|_1 = \sum_{i,j} |y_{ij}|$  is a convex relaxation of  $\|\cdot\|_0$ ,  $\lambda$  is a tuning parameter to balance data fitting and sparsity level. OLM alternatively updates  $\mathbf{Y}$  and  $\mathbf{D}$  as follows. When  $\mathbf{D}$  is fixed, it randomly picks a batch of columns of  $\mathbf{X}$  and applies sparse coding to each selected column. Letting  $S$  be the index set of all previously selected samples and  $\mathbf{Y}_S$  contain their sparse coefficients, the method then updates  $\mathbf{D}$  to the solution of

$$\min_D \{\|DY_S - X_S\|_F^2, \|d_i\|_2 \leq 1, \forall i\} \quad (4.13)$$

Where  $X_S$  denotes the submatrix consisting of all columns of  $X$  indexed by  $S$ . The above two steps are then repeated until convergence. The algorithm often runs faster than KSVD, and its efficiency relies on the assumption that all training samples have the same distribution. Assuming that the training data admits bounded probability with a compact support and  $Y_S Y_S^T$  is uniformly positive definite, it is shown that the iterate sequence asymptotically satisfies the first-order optimality condition of (1.3). The global convergence of the iterate sequence is still open [3, 4].

Most updated dictionary learning [15] algorithms are second-order iterative batch procedures, accessing the whole training set at every iteration to minimize the cost function under some constraints. They show experimentally much faster than first order gradient descent method [28] but unable to handle very large training sets and dynamic training data which changes over the time like video sequence. To handle this problem we need to use online approach each element at a time. This is very important for video and image processing [29], it is usual to learn dictionary adapted to small patches, with the training data, which generally include several millions of these patches, which is roughly one per frame. By using this process, online method is based on stochastic approximation is an attractive alternative to batch methods. First order stochastic gradient descent with projections on the constraint set is sometimes used for dictionary learning. But we can go beyond and develop a sparse coding structure to design optimize method to dictionary learning problem having low memory consumption and less computational cost than classical second-order batch algorithms and without the need of explicit learning rate tuning.

The dictionary method that is presented in the next section is one type of online algorithms based on stochastic approximations; one sample is processed at a time, but exploits the specific structure of the problem to efficiently solve it. Contrary to classical first-order stochastic gradient descent, it does not require explicit learning rate tuning and minimizes a sequentially quadratic local approximation of the expected cost [15].

#### **Online dictionary learning algorithm [15]:**

Here Algorithm 2 and 3 is going to be represented and simultaneously we will show two variants that will make the process faster.

---

**Algorithm 2: Online dictionary learning**

---

Let,  $\mathbf{x} \in \mathbb{R}^m \sim \mathcal{P}(\mathbf{x})$ ,

$\mathcal{P}(\mathbf{x})$  is a random variable and this algorithm to draw i.i.d samples of  $\mathcal{P}$ ,  $\lambda \in \mathbb{R}$  (regularization parameter),  $\mathbf{D}_0 \in \mathbb{R}^{m \times k}$ , (initial dictionary) and T is number of iteration

- $\mathbf{A}_0 \leftarrow 0$ ,  $\mathbf{B}_0 \leftarrow 0$  (resetting past informations)
- for  $t=1$  to T do
- Draw  $\mathbf{x}_t$  from  $\mathcal{P}(\mathbf{x})$
- Sparse coding: using LARS

$$\boldsymbol{\alpha}_t = \underset{\boldsymbol{\alpha} \in \mathbb{R}^k}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{x}_t - \mathbf{D}_{t-1} \boldsymbol{\alpha}\|_2^2 + \lambda \|\boldsymbol{\alpha}\|_1$$

- $\mathbf{A}_t \leftarrow \mathbf{A}_{t-1} + \boldsymbol{\alpha}_t \boldsymbol{\alpha}_t^T$
- $\mathbf{B}_t \leftarrow \mathbf{B}_{t-1} + \mathbf{x}_t \boldsymbol{\alpha}_t^T$
- Compute  $\mathbf{D}_t$  using algorithm 2, with  $\mathbf{D}_{t-1}$  as warm reset, so that

$$\begin{aligned} \mathbf{D}_t &= \underset{\mathbf{D} \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \sum_{i=1}^t \frac{1}{2} \|\mathbf{x}_i - \mathbf{D} \boldsymbol{\alpha}_i\|_2^2 + \lambda \|\boldsymbol{\alpha}_i\|_1, \\ &= \underset{\mathbf{D} \in \mathcal{C}}{\operatorname{argmin}} \frac{1}{t} \left( \frac{1}{2T_r(\mathbf{D}^T \mathbf{D} \mathbf{A}_t)} - T_r(\mathbf{D}^T \mathbf{B}_t) \right) \end{aligned}$$

- End for
  - Return  $\mathbf{D}_t$  (learned dictionary).
- 

---

**Algorithm 3: Dictionary Update**

---

Let,

$\mathbf{D} = [\mathbf{d}_1, \dots, \mathbf{d}_k] \in \mathbb{R}^{m \times k}$ , input dictionary,

$$\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_k] \in \mathbb{R}^{k \times k} = \sum_{i=1}^t \boldsymbol{\alpha}_i \boldsymbol{\alpha}_i^T,$$

$$\mathbf{B} = [\mathbf{b}_1, \dots, \mathbf{b}_k] \in \mathbb{R}^{m \times k} = \sum_{i=1}^t \mathbf{x}_i \boldsymbol{\alpha}_i^T,$$

- repeat
- for  $j = 1$  to  $k$  do
- update the  $j$ -th column to optimize for eq. 9

$$\mathbf{u}_j \leftarrow \frac{1}{A_{jj}} (\mathbf{b}_j - \mathbf{D} \mathbf{a}_j) + \mathbf{d}_j$$

$$\mathbf{d}_j \leftarrow \frac{1}{\max(\|\mathbf{u}_j\|_2, 1)} \mathbf{u}_j$$

- end for
- until convergence

Return  $\mathbf{D}$  (updated dictionary)

---

#### 4.4. BLOCK PROXIMAL GRADIENT METHOD:

Here[16] considers regularized block multi-convex optimization for solving this problem and proposed a Block proximal gradient(BPG) method, where objective function and the feasible set are generally non-convex but convex in block of variables. Here propose a generalized block coordinate descent method. The proposed algorithms are adapted to factorize nonnegative matrices and tensors. As well as for completing them from their incomplete observations. On synthetic data the algorithms were tested. Compared to the existing other dictionary learning algorithms, the proposed algorithms demonstrate to superior performance in both solution and speed quality.

Consider,

$$\min_{\mathbf{x}, \mathbf{y}} f(\mathbf{x}, \mathbf{y}) + r_x(\mathbf{x}) + r_y(\mathbf{y}) \quad (4.14)$$

Where  $f$  is convex and differentiable with respect to either  $\mathbf{y}$  or  $\mathbf{x}$  by fixing the other one. Here  $r_x$  and  $r_y$  are extended-valued convex functions. After the  $k$ -th iteration of BPG,  $\mathbf{x}$  and  $\mathbf{y}$  are alternatively updated by

$$\mathbf{x}^k = \underset{\mathbf{x}}{\operatorname{argmin}} \langle \nabla_{\mathbf{x}} f(\hat{\mathbf{x}}^k, \mathbf{y}^{k-1}), \mathbf{x} - \hat{\mathbf{x}}^k \rangle + \frac{L_x^k}{2} \|\mathbf{x} - \hat{\mathbf{x}}^k\|_2^2 + r_x(\mathbf{x}), \quad (4.15a)$$

$$\mathbf{y}^k = \underset{\mathbf{y}}{\operatorname{argmin}} \langle \nabla_{\mathbf{y}} f(\mathbf{x}^k, \hat{\mathbf{y}}^{k-1}), \mathbf{y} - \hat{\mathbf{y}}^k \rangle + \frac{L_y^k}{2} \|\mathbf{y} - \hat{\mathbf{y}}^k\|_2^2 + r_y(\mathbf{y}), \quad (4.15b)$$



Here, with respect to

$$\mathbf{x}, \hat{\mathbf{x}}^k = \mathbf{x}^{k-1} + \omega_x^k(\mathbf{x}^{k-1}, \mathbf{x}^{k-2}),$$

$L_x^k$  is a Lipschitz constant of  $\nabla_{\mathbf{x}} f(\mathbf{x}, \mathbf{y}^{k-1})$ , which denotes an extrapolated point with weight  $\omega_x^k \geq 0$ , and  $L_y^k$  and  $\hat{\mathbf{y}}^k$  have the same meanings for  $\mathbf{y}$ .

BPG is a variant of the block coordinate minimization (BCM) method [19], which updates cyclically  $\mathbf{x}$ ,  $\mathbf{y}$ . BPG minimize the objective with respect to one block of variables at a time, while the other one is fixed at its most present value. To have closed form solutions, the update of  $\mathbf{r}_x$  and  $\mathbf{r}_y$  is essential, because the BCM decreases the objective faster, and sub problems for BCM are usually much more difficult than those in (4.15).

### Learning Method [3, 4]:

We learn this dictionary from training dataset  $\mathbf{X}$  via solving (4.12). Let

$$\ell(\mathbf{D}, \mathbf{Y}) = \frac{1}{2} \|\mathbf{D}\mathbf{Y} - \mathbf{X}\|_F^2$$

be the fidelity term in (4.12). Applying (4.15) to (4.12), we can alternatively update  $\mathbf{D}$  and  $\mathbf{Y}$  by

$$\mathbf{D}^K = \min_{\mathbf{D} \in \mathcal{D}} \langle \nabla_{\mathbf{D}} \ell(\hat{\mathbf{D}}^k, \mathbf{Y}^{k-1}), \mathbf{D} - \hat{\mathbf{D}}^k \rangle + \frac{L_d^k}{2} \|\mathbf{D} - \hat{\mathbf{D}}^k\|_F^2 \quad (4.16a)$$

$$\mathbf{Y}^K = \min_{\mathbf{Y}} \langle \nabla_{\mathbf{Y}} \ell(\hat{\mathbf{D}}^k, \mathbf{Y}^k), \mathbf{Y} - \hat{\mathbf{Y}}^k \rangle + \frac{L_y^k}{2} \|\mathbf{Y} - \hat{\mathbf{Y}}^k\|_F^2 + \lambda \|\mathbf{Y}\|_1 \quad (4.16b)$$

Where,

$$\mathcal{D} = \{\mathbf{D}: \|\mathbf{d}_i\|_2 \leq 1, i = 1, \dots, K\} \text{ is the limited set of } \mathbf{D}.$$

The updated form of (4.16) can be written as

$$\mathbf{D}^K = \mathcal{P}_{\mathcal{D}}(\hat{\mathbf{D}}^k - \frac{1}{L_d^k} \nabla_{\mathbf{D}} \ell(\hat{\mathbf{D}}^k, \mathbf{Y}^{K-1})), \quad (4.17a)$$

$$\mathbf{Y}^K = S_{\lambda/L_y^k}(\hat{\mathbf{Y}}^k - \frac{1}{L_y^k} \nabla_{\mathbf{Y}} \ell(\mathbf{D}^k, \hat{\mathbf{Y}}^k)), \quad (4.17b)$$

In (4.17a)  $\mathcal{P}_{\mathcal{D}}(\cdot)$  is the Euclidean projection to  $\mathcal{D}$  defined for any  $\mathbf{D}$  as

$$(\mathcal{P}_{\mathcal{D}}(\mathbf{D})) = \mathbf{d}_i / \max(1, \|\mathbf{d}_i\|), i = 1, \dots, K \quad (4.18)$$

and in (4.17b),  $S_{\tau}(\cdot)$  here denotes soft-thresholding operator defined for any  $\mathbf{Y}$  by

$$(S_{\tau}(\mathbf{Y}))_{i,j} = \text{sign}(y_{i,j}) \cdot \max(|y_{i,j}| - \tau, 0), \forall i, j \quad (4.19)$$

Here above throughout the numerical representation,

$$L_d^k = \|\mathbf{Y}^{K-1}(\mathbf{Y}^{K-1})^T\|, L_y^k = \|\mathbf{D}^K(\mathbf{D}^K)^T\| \quad (4.20)$$

And the weights of the above are

$$\omega_x^k = 0.9999 \min(\omega^k, \sqrt{\frac{L_d^{k-1}}{L_d^k}}), \quad (4.21)$$

$$\omega_y^k = 0.9999 \min(\omega^k, \sqrt{\frac{L_y^{k-1}}{L_y^k}}) \quad (4.22)$$

To make the whole process speed up, the assumption of weights is helpful. To perform stable and converge rapidly and the non-increasing property is both required and also important. The Algorithm 4 of the BPG method is described below [3, 4]-

---

**Algorithm 4: Block proximal gradient for dictionary learning**

---

**Data:** training samples  $\mathbf{X}$ , parameter  $\lambda > 0$ , and initial points  $(\mathbf{D}^{-1}, \mathbf{Y}^{-1}) = (\mathbf{D}^0, \mathbf{Y}^0)$

**for**  $k = 1, 2, \dots$  **do**

- Set  $L_d^k$  and  $\omega_d^k$  by (4.20) and (4.21, 4.22), respectively
- Let  $\hat{\mathbf{D}}^k = \mathbf{D}^{K-1} + \omega_d^k(\mathbf{D}^{K-1} - \mathbf{D}^{K-2})$  and get  $\mathbf{D}^K$  by (4.17a).
- Set  $L_y^k$  and  $\omega_y^k$  by (4.20) and (4.21, 4.22), respectively.
- Let  $\hat{\mathbf{Y}}^k = \mathbf{Y}^{K-1} + \omega_y^k(\mathbf{Y}^{K-1} - \mathbf{Y}^{K-2})$  and get  $\mathbf{Y}^K$  by (4.17b).

**if**  $F(\mathbf{D}^K, \mathbf{Y}^K) > F(\mathbf{D}^{K-1}, \mathbf{Y}^{K-1})$  **then**

Re-update  $\mathbf{D}^K$  and  $\mathbf{Y}^K$  by (4.17a) and (4.17b) with  $\hat{\mathbf{D}}^k = \mathbf{D}^{K-1}$  and  $\hat{\mathbf{Y}}^k = \mathbf{Y}^{K-1}$ , respectively.

**if** Some stopping conditions are satisfied **then**

Output  $(\mathbf{D}^K, \mathbf{Y}^K)$  and stop.

---

This algorithm uses adjoining update for both  $\mathbf{Y}$  and  $\mathbf{D}$ . The difference from other methods such as KSVD and OLM that propose exact minimization to update  $\mathbf{D}$  and/or  $\mathbf{Y}$ . To maintain the closed form solutions for both  $\mathbf{D}$  and  $\mathbf{Y}$ -sub problems ensures that, the algorithm have the extrapolated technique and a lower per-iteration complexity, which help it take a small number of iterations to get a faithful solution[3,4].

#### ***4.5 CONCLUSION:***

Dictionary learning has very popularly applied to image denoising, super-resolution, classification problem and feature extraction. Three dictionary learning algorithm has drawn in this chapter. KSVD and OLM are two very popular dictionary learning algorithm in image processing domain. BPG method also described here. The deblurring technique is based on this three method in this thesis.

# CHAPTER-5

## *IMAGE DEBLURRING WITH PATCH DICTIONARY METHOD*

- INTRODUCTION
- IMAGE DEBLURRING
- THE MODEL
- PROCEDURE OF PATCH DICTIONARY METHOD

# CHAPTER-5

## 5.1. INTRODUCTION:

Here in the image deblurring problem, the general issue is to restore a blurred image, which is corrupted by zero-mean white and homogeneous Gaussian additive noise. Over trained dictionary the approach has taken based on sparse and redundant representations. Among various algorithm effectively three patched based dictionary learning methods are used to solve this problem. In patched based method there are some issue for overlapping patches because for overlapping excessive number of dictionary coefficients have to determine. It also be very ineffective to update a few patches at a time. The focus is to divide it in multiple sub problems. The strategy is to divide the whole image in a subset of non-overlapping patches, which can handle the sparse recovery for each subset. Then to get the final recovery the sub problems are averaged. This method is very effective for both time and quality. Additionally three different dictionary learning method is used for comparative analysis of quality. Also weight variation is another comparison analysis through the study of PSNR value. Here proposed an intelligent way to choose the weight for different blurring artifacts and noise ratio. In [3, 4] proposed the whole image recovery by patched based dictionary learning method. In this work only image deblurring procedure will be discussed with the help of matlab code in [2, 3, 4].

In [3, 4]  $\mathbf{M}$  is an image, corrupted by noise in the form of  $\mathbf{b} = \mathbf{A}(\mathbf{M}) + \xi$ , where  $\mathbf{A}$  is a linear operator and  $\xi$  is some noise. For different application  $\mathbf{A}$  is used as different operator like recovery include image denoising ( $\mathbf{A}$  equals to the identity operator  $\mathbf{I}$ ), super-resolution ( $\mathbf{A}$  is a down sampling operator), compressive imaging recovery ( $\mathbf{A}$  is compressed sensing operator), image deblurring ( $\mathbf{A}$  is a blurring operator), as well as medical imaging recovery ( $\mathbf{A}$  can be down sampled Fourier or Radon operators, as for example). This work restore the blur image to  $\mathbf{M}$  by computing sparse representation under a learned dictionary. Here numerically study the dictionaries that sparsely represents each and all the overlapping patches of  $\mathbf{M}$ . But the problem is large number of overlapping patches lead to a large number of free coefficients in the recovery. It can cause slow computation and overfitting issue. To avoid the overfitting issue, have to process one subset of non-overlapping, covering patches at a time. (Covering means the subset of patches covers all of the pixels of the image.) It is consider the more difficult “global” kind of task. Here the process is

subset of patches are sparsely solved each and obtain a recovery of the whole image. After that have to process multiple different subsets of covering, non-overlapping patches, we obtain multiple whole-image recovery. To eliminate the grid artifact average is taken that might exist in the individual ones. For each subset,  $8 \times 8$  patches are used in [3, 4]. Here two types of patch size in this work-  $8 \times 8$  and  $6 \times 6$ . The averaged recovery has relatively a higher PSNR than other state of approaches which address the overfitting issue, by applying either online optimization or incorporating additional image structures. Dictionary is pre-learned from a set of similar images, and then either fixed during the recovery or iteratively updated in adaptive under image recovery. Following [5], after recovering an image, we update the dictionary to fit the recovered image by solving an  $\ell_1$ -regularized model. Now the detail model of this study is in next.

## 5.2. IMAGE DEBLURRING:

Various different methods have been developed now a day to restore an image from its corrupted and/or incomplete measurement. One of the popular class of recovery methods are based on sparse coding and dictionary learning such as those in [5, 12, 13]. A signal  $\mathbf{x} \in \mathbb{R}^n$  is sparse (or approximately sparse) under a dictionary  $\mathbf{D} \in \mathbb{R}^{n \times K}$ , when  $\mathbf{x} = \mathbf{D}\mathbf{y}$  (or  $\mathbf{x} \approx \mathbf{D}\mathbf{y}$ ) and  $\mathbf{y} \in \mathbb{R}^K$  has only a few nonzero. Some particular dictionary can sparsely represent various types of signals. For example, under dictionaries based on various wavelet, curvelets, shearlet, and other transforms natural images are approximately sparse. Suppose  $\mathbf{x}$  has a sparse representation under a dictionary  $\mathbf{D}$ . Then given  $\mathbf{D}$  and linear measurements  $\mathbf{b} = \mathbf{A}(\mathbf{x}) + \boldsymbol{\xi}$ , one can recover  $\mathbf{x}$  through sparsely coding  $\mathbf{x}$  via solving

$$\min_{\mathbf{y}} \|\mathbf{y}\|_0, \text{ s.t. } \|\mathcal{A}(\mathbf{D}\mathbf{y}) - \mathbf{b}\|_2^2 \leq \epsilon \quad (5.1)$$

The counts of nonzero number represents as  $\|\cdot\|_0$  which approximated by  $\|\cdot\|_1$  for tractable computation.  $\epsilon \geq 0$  corresponds to  $\boldsymbol{\xi}$ . The original signal  $\mathbf{x}$  can be estimated by  $\mathbf{D}\mathbf{y}$  by the solution  $\mathbf{y}$  of (5.1). Dictionary can be choose in two way-either predetermined or learned from a set of training data. Orthogonal or over complete wavelets, curvelets, and discrete cosine transforms (DCT) are predetermined dictionaries have better analytical and numerical properties than a learned one. But a learned dictionary can better adapt to natural signals and improve the recovery quality of image. Existing methods such as KSVD [14], OLM [15], BPG [16, 3, 4] learn a

dictionary  $\mathbf{D}$  to sparsely represent of the patches of an image, rather than the whole image itself. The size of the patches can be  $6 \times 6$  or  $8 \times 8$ . In the work [5] overlapping patches of  $\mathbf{M}$  are denoised by a patch size dictionary and sparse coding. Then averaging technique will be performed. But this method is simple and can include additional energy terms and constraints, as well as to be embedded in more complicated image processing applications.

### 5.3. THE MODEL:

Following the approach described in [7] this image can be represent:

$$\mathbf{M} = (\mathcal{T}_P)^{-1}(\sum_{(i,j) \in P} \mathcal{R}_{ij}^T \mathcal{R}_{ij}(\mathbf{M})), \quad \mathcal{T}_P = \sum_{(i,j) \in P} \mathcal{R}_{ij}^T \mathcal{R}_{ij} \quad (5.2)$$

Where for  $(i, j)$ -th patch,  $\mathcal{R}_{ij}$  is an operator. The adjoint of  $\mathcal{R}_{ij}$  is  $\mathcal{R}_{ij}^T$ .  $P$  refers a subset of patches, which covers the all pixel of  $\mathbf{M}$ . Also  $\mathcal{T}_P$  is invertible.  $\mathcal{T}_P$  is diagonal, in a pixel by pixel manner its inverse would be implemented. If  $P$  has a particular sparse representation for every patch  $\mathcal{R}_{ij}(\mathbf{M})$  under  $\mathbf{D}$ , then for a sparse vector  $\mathbf{y}_{ij}$ ,  $\mathcal{R}_{ij}(\mathbf{M}) = \mathbf{D}\mathbf{y}_{ij}$ . Then the above representation can be written as:

$$\mathbf{M} = (\mathcal{T}_P)^{-1}(\sum_{(i,j) \in P} \mathcal{R}_{ij}^T(\mathbf{D}\mathbf{y}_{ij})) \quad (5.3)$$

The following  $\ell_1$  model can be written using this representation:

$$\min_{\mathbf{y}} \sum_{(i,j) \in P} \|\mathbf{w}_{ij} \odot \mathbf{y}_{ij}\|_1, \text{ s.t. } \|\mathcal{A}\mathcal{T}_P^{-1}(\sum_{(i,j) \in P} \mathcal{R}_{ij}^T(\mathbf{D}\mathbf{y}_{ij})) - \mathbf{b}\|_2 \leq \sigma \quad (5.4)$$

Here  $\sigma$  is the noise representation determined by,  $\mathbf{w}_{ij} \geq \mathbf{0}$  is a weight vector, and “ $\odot$ ” denotes

Component wise product. The below model can be considered equivalently:

$$\min_{\mathbf{y}} \sum_{(i,j) \in P} \|\mathbf{w}_{ij} \odot \mathbf{y}_{ij}\|_1 + \frac{1}{2v} \|\mathcal{A}\mathcal{T}_P^{-1}(\sum_{(i,j) \in P} \mathcal{R}_{ij}^T(\mathbf{D}\mathbf{y}_{ij})) - \mathbf{b}\|_2^2 \quad (5.5)$$

Here  $v$  is the parameter corresponding to  $\sigma$ .

By solving (5.4) and (5.5) the following equation is proposed:

$$\mathcal{T}_P^{-1} \sum_{(i,j) \in P} \mathcal{R}_{ij}^T(\mathbf{D}\mathbf{y}_{ij}) . \text{ This equation can be used to estimate the } \mathbf{M}.$$

### Choice of P:

P is the subset of covering patches. In [5] using all the overlapping patches for  $\mathcal{L}$ , too many unknowns introduces. To recover an s-sparse signal of length n, the  $\ell_1$  minimization typically needs  $O(s \log(n/s))$ . Suppose each patch corresponds  $y_{ij}$ . It has at least r nonzero and all the overlapping patches  $(N_1 - n_1 + 1)(N_2 - n_2 + 1)$  are used. Y has  $r(N_1 - n_1 + 1)(N_2 - n_2 + 1)$  nonzero out of  $n = K(N_1 - n_1 + 1)(N_2 - n_2 + 1)$  entries. We have at most  $N_1 N_2$  measurements which is not sufficiently many to reach  $O(s \log(n/s)) = O(r N_1 N_2 \log(K/r))$ . So all the patches cannot be used. The subset of covering patches P, now let it be non-overlapping patches [3, 4]. Now setting  $\mathcal{A} = \mathcal{L}$ , and  $\mathbf{b} = \mathbf{M} + 0.05\boldsymbol{\xi}$  (5.6) with  $\boldsymbol{\xi} \sim \mathcal{N}(0, I)$  the process is derived. In [3, 4] two process is compared for two different P's. The result is depend upon PSNR value of the two resultant image. Using all overlapping patches, the PSNR value of the resultant image is less than the PSNR value of non-overlapping patch based resultant image. So using all overlapping patches is worst result that the other.

The model is similar as [6]

$$\min_y \sum_{(i,j) \in S} \|y_{ij}\|_1 \frac{1}{2v} \left\| \mathcal{A} \left( \left( \sum_{(i,j) \in S} \mathfrak{R}_{ij}^T \mathfrak{R}_{i,j} \right)^{-1} \left( \sum_{(i,j) \in S} \mathfrak{R}_{ij}^T (\mathbf{D}_{k_{ij}} y_{ij}) \right) - \mathbf{b} \right) \right\|_2^2 + AR(y) + NLS(y) \quad (5.7)$$

Where S is the set of all overlapping patches, v is a balancing sparsity parameter.  $\mathbf{D}_{k_{ij}}$  is a local dictionary, which is used to represent the (i, j)-th patch. AR (·) and NLS (·) are two regularization terms, which corresponds to local auto-regression and non-local similarity. To reduce variable freedom and increase recoverability of (5.6), AR and NLS terms are used.

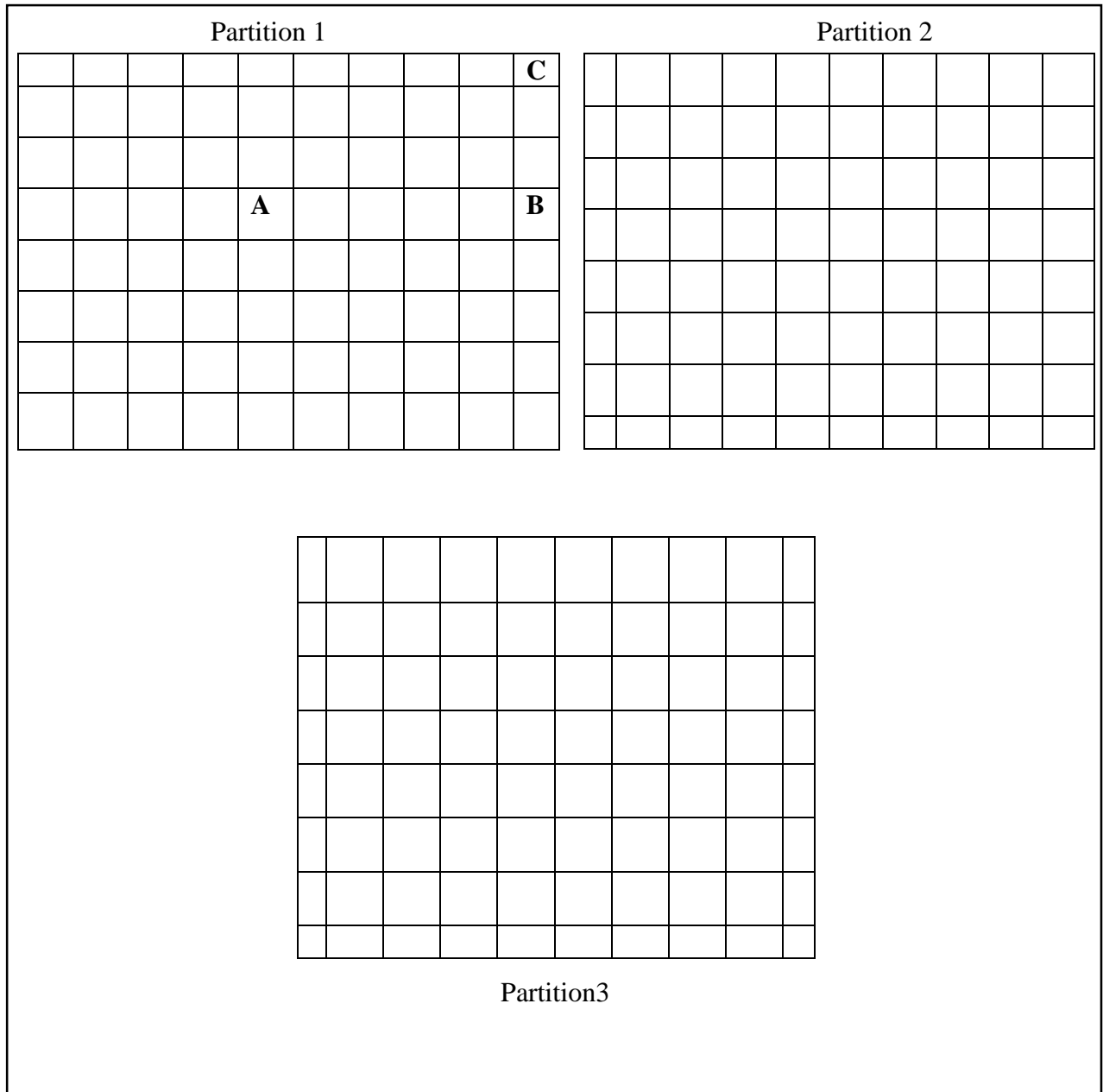
There is an another problem that is the image cannot divide evenly in same patches to cover the whole image. So some technique have introduced to allow the P in different sizes. In this technique P can also contain some smaller patches near the boundary. Now the technique represent partition of the image arbitrarily with blocks no greater than  $n_1 \times n_2$ . The techniques are given below:

### Assumptions techniques:

The way described in [3, 4] by which an image is partitioned into patches is uniquely determined. All interior patches are  $8 \times 8$ . But the corner patches alignment are different-



- Right and upper boundary patches are not  $8 \times 8$ . In partition 1(5.1), all the right boundary patches are  $8 \times 4$ , and the upper-right corner patch is  $4 \times 4$
- Left and lower boundary patches are not  $8 \times 8$ . In partition 2(5.1), all the left boundary patches are  $8 \times 4$ , and the lower-left corner patch is  $4 \times 4$ .
- In partition 3 (5.1) all the left and right boundary patches are  $8 \times 2$ , and the lower-left and lower-right corner patches are  $4 \times 2$ .



*Fig.5.1. Image partition by patches [3, 4]*

#### 5.4. PROCEDURE OF PATCH DICTIONARY METHOD [3, 4]:

P define non-overlapping covering patches, then each pixel must be contained by exactly one patch. To verify  $\mathcal{T}_P = \mathcal{L}$  is not difficult. If  $(i, j) \in P$  is one interior patch, so that  $\mathcal{R}_{ij}(\mathbf{M})$  means to take, the  $(i, j)$ -th patch of  $\mathbf{M}$ , and  $\mathcal{R}_{ij}^T$  is to first generate an  $N_1 \times N_2$  zero matrix, and then add  $\mathbf{x}$  to its  $(i, j)$ -th patch. Need to act accordingly, however, as  $(i, j) \in P$  is a boundary or corner patch and the size of the patch is smaller than  $n_1 \times n_2$ . Suppose, let  $(i, j)$  be patch “C” in Figure (5.1).

##### Variable:

- $\mathcal{R}_{ij}(\mathbf{M})$ : First generate an  $8 \times 8$  zero matrix, and then replace with its upper-right  $4 \times 4$  corner submatrix with the upper-right  $4 \times 4$  corner patch of  $\mathbf{M}$ ;
- $\mathcal{R}_{ij}^T(\mathbf{x})$ : First generate a  $100 \times 100$  zero matrix, and then replace with the upper-right  $4 \times 4$  corner patch corresponding to “C” with the upper-right  $4 \times 4$  corner submatrix of  $\mathbf{x}$ .

Here observed that with non-overlapping patches there are some visible artifacts on block boundaries when solving (5.5). For this problem good recovery does not generate. So the equation (5.5) is calculated for different P parallel. The procedure is executed for three type partitioned image in fig 5.1. For each Partition the (5.5) have to be solved. Then the averaging technique have performed for different P's. By this process the PSNR value improves. The whole procedure is summarized in Algorithm 1[3, 4].

---

#### Algorithm 1:

---

**Data:** Dictionary  $\mathbf{D}$ , image size  $(N_1, N_2)$ , patch size  $(n_1, n_2)$ , measurements  $\mathbf{b}$ , linear operator  $\mathcal{A}$ , and parameter  $v$ .

**Choose**  $t$  in different ways to partition the whole image into non-overlapping patches; denote them as  $P_1, \dots, P_t$ .

**Solve** (5.5) for  $P_k$  and let the recovered image be  $\mathbf{M}_k$ , for  $k=1 \dots t$ .

**Average** all the recovered images by  $\tilde{\mathbf{M}} = \frac{1}{t} \sum_{k=1}^t \mathbf{M}_k$  and output is  $\tilde{\mathbf{M}}$ .

---

**Dictionary update:**

Dictionary have now updated using the patches extracted from  $\tilde{M}$  , which is estimated from Algorithm 1. The updated dictionary should be better because  $\tilde{M}$  is close to the original image  $M$ . This updated procedure have processed for several times to get better result. In [2, 3] after one adaptive update the change is significant, not more better for next process. The adaptive method summarizes in Algorithm 2.

---

**Algorithm 2:**

---

**Data:** Dictionary  $D$ , image size  $(N1, N2)$ , patch size  $(n1, n2)$ , measurements  $\mathbf{b}$ , linear operator  $\mathcal{A}$ , and parameter  $v$ .

**Repeat**

**Run** Algorithm 1 to get the recovered image  $\tilde{M}$ .

**Update** the dictionary  $D$  from the extracted patches from  $\tilde{M}$ .

**Until** convergence.

---

In this section the whole patched based dictionary method has discussed here. The whole procedure is proposed by two algorithm 1 and 2. This method of image deblurring is used for medical image recovery. In this thesis work, MRI and CT scan image of brain has been studied through the help of this patched based dictionary method [3, 4]. This study and comparative results for different variables with graph has described in next module. Also some details of implementation of this process is in next chapter.

# CHAPTER-6

## *EXPERIMENTAL STUDY*

- INTRODUCTION
- IMPLEMENTATION
- EXPERIMENTAL TEST
- PERFORMANCES ANALYSIS AND COMPARISON OF VARIOUS RESULT
- INTELLIGENT METHOD
- CONCLUSION

# CHAPTER-6

## 6.1 INTRODUCTION:

In this section the study has been proposed on ten MRI and ten CT scan of brain images which is taken from [9, 10]. Then the images have blurred by using two types of blurring process. One is average blur and another one is motion blur. In this work, the study is on both two blurring techniques. Also three different dictionary learning method have proposed here. The whole deblurring process on based on three different dictionaries have compared on the basis on PSNR value. The quality of restored images depends upon another variable, that is weight variable. In this thesis, work an intelligent way has learned which propose best-recovered image based on different weight. In addition, another variable is noise percentage. Here is a comparative study for different percentage of noise. The intelligent representation way to choose best PSNR value automatically has recorded in Table 6.31 and 6.32. in below.

## 6.2 IMPLEMENTATION:

In (5.7)  $\mathbf{b} = \mathbf{A}(\mathbf{M}) + \sigma \xi$ , where  $\xi \sim \mathcal{N}(0, I)$  is Gaussian noise, and for noise the equation is  $\sigma = n \|\mathcal{A}(\mathbf{M})\|_2 / \|\xi\|_2$ , where the value of  $n$  is 0.01 and 0.05 throughout the tests.  $v = 0.1\sigma$  for the third kind of  $\mathbf{A}$  which is blurring. Additionally all elements of  $w_{ij}$  is one except its first component, which was set to zero. For this setting, using any DC as the first atom of  $\mathbf{D}$  would make no difference for the solution of (5.5). Then, (5.5) has solved via YALL1 (version 1.4) [11]. Here used Gaussian random starting point and  $10^{-4}$  as its stopping tolerance as in [2]. All other parameters of YALL1 were set to their default values in [3, 4]. YALL1 has chosen due to its high efficiency for solving (5.5). Moreover, it is easy to call by providing operations of  $\mathcal{A}$  and  $\mathcal{A}^T$ .

Here consider  $\mathbf{A}$  was a blurring operator with a  $9 \times 9$  kernel, `fspecial('average', [9, 9])` and `fspecial('motion', 10, 45)` respectively used for two different kernels, which were generated by Matlab's commands available in [2, 3, 4]. This method identically have chosen from the works of Xu and Yin [3, 4]. One FFT, one inverse FFT, and some component wise products can also realize the implementation of a blurring operator. Hence, the experiment with variation of different operator has described below.

### 6.3 EXPERIMENTAL TEST:

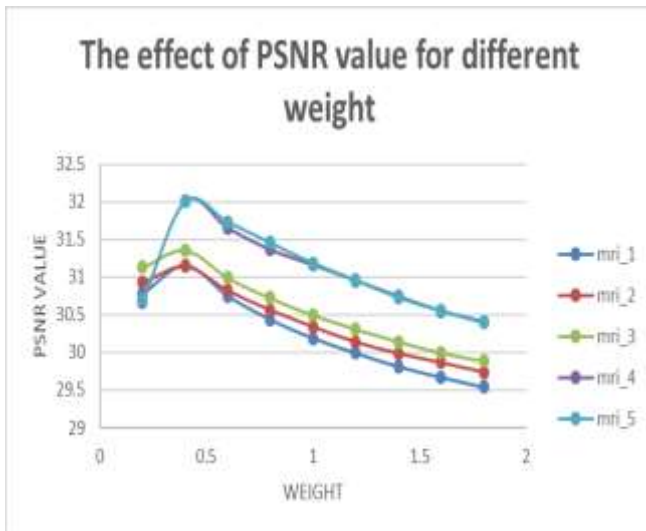
This test compares three methods – BPG, KSVD and OLM for both MRI and CT scan images. A dictionary  $\mathbf{D} \in \mathbb{R}^{n \times k}$  is generated with the Matlab command `randn(n,K)`. Then each column of  $\mathbf{D}$  has normalized to have unit  $\ell_2$ -norm. Then  $p$  training samples have generated in the  $n$ -dimensional space. Each sample is a linear combination  $r$  columns of  $\mathbf{D}$ , and the coefficients of  $\mathbf{D}$  were Gaussian randomly generated. In (4.4)  $T_0$  is set by  $r$  which is the sparsity level and in (4.12)  $\lambda = 0.5/\sqrt{n}$  is obtained. All other parameter have varied for each dictionary of variation of sparsity level. The weight vector in (5.4) has changed from 0.2 to 1.8. For this variation, the PSNR value has changed accordingly. This variation has charted below for different dictionary of different sparsity level in Table 6.1 to 6.12 and 6.15 to 6.26. In (5.6) a noise variation has denoted by  $\xi$ . This parameter has changed in two variation. 1% and 5% noise have applied here. For 1% noise the chart has proposed in Table 6.1 to 6.12. In addition, for 5% noise the chart has listed in Table 6.15 to 6.26. The variation of PSNR value of the chart has plotted correspondingly. For each Table two graph have plotted. In the work ten MRI and ten CT scan images have, taken from [9, 10]. That's why for each Table first table is for first five images, and next is for next five images. Following [3, 4] the blurring technique is two. One is average blur and another one is motion blur. The Table 6.13 to 6.14 and Table 6.27 to 6.32 has provided for comparison of views of the MRI and CT scan images for different blur technique and also for two-noise variation 1% and 5%. This work provide an intelligent process to choose the best PSNR value automatically for different weight variation when other parameter has default values. This process description has objected in Table 6.33.

### 6.4 PERFORMANCE ANALYSIS AND COMPARISON OF VARIOUS RESULTS:

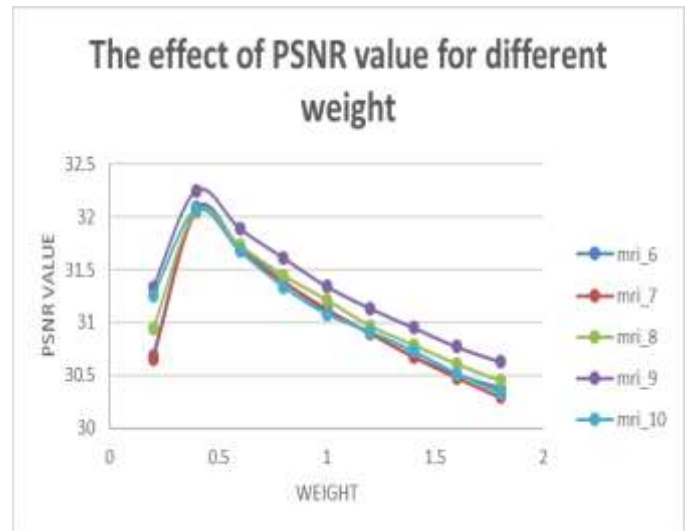
The comparison study of deblurring performances has varied for different variables. All the performances are provided in Table and Graph format. The provided data is present for two value of  $\lambda$  in BPG method, two values of  $r$  (sparsity) in KSVD method and two variation 16 and 32 of OLM. Here N1 presents 1% noise and N2 presents 2% noise. Through the extended study of [3, 4], comparative results are shown below. The ten MRI and CT scan images are named thoroughly as mri\_1, mri\_2 ... so on, and ct\_1, ct\_2 ... so on.

MRI_BPG_λ8_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	30.7762	30.9281	31.132	30.6674	30.7066	30.6909	30.652	30.9412	31.3376	31.2577
0.4	31.1518	31.1487	31.3638	32.0162	32.0044	32.0868	32.0517	32.0515	32.2476	32.0682
0.6	30.7483	30.8256	30.9935	31.6554	31.735	31.7225	31.6854	31.7308	31.8888	31.6769
0.8	30.4353	30.5663	30.7246	31.3679	31.4586	31.3918	31.3925	31.4477	31.6144	31.339
1	30.1909	30.3437	30.4992	31.1652	31.189	31.1087	31.1288	31.2155	31.3442	31.0805
1.2	29.9941	30.1409	30.3138	30.9572	30.9618	30.8986	30.901	30.9712	31.1356	30.9112
1.4	29.8172	29.9893	30.1456	30.7467	30.7341	30.6758	30.6757	30.7889	30.9544	30.7266
1.6	29.6746	29.8676	29.9979	30.551	30.5564	30.5044	30.4764	30.6103	30.7729	30.5131
1.8	29.5445	29.7404	29.8872	30.4088	30.3991	30.3773	30.2917	30.4503	30.631	30.333

*Table.6.1.PSNR Value chart for MRI images – BPG ( $\lambda=0.8$ ) (1% noise)-average blur*



*Fig.6.1. Variation in PSNR performance for different values of w*



*Fig.6.2. Variation in PSNR performance for different values of w*

CT_BPG_λ8_N1										
WEIGHT	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.3004	27.6553	26.6644	26.0081	27.0753	27.7056	25.6688	26.9717	27.8507	28.2023
0.4	27.9231	27.2578	26.6138	25.8391	26.9193	27.2014	25.6851	26.5586	27.4154	27.5282
0.6	27.5234	26.7784	26.183	25.4566	26.4788	26.7199	25.2728	26.0948	26.9771	27.0403
0.8	27.1703	26.3927	25.8283	25.1051	26.1543	26.3653	24.9125	25.6766	26.6384	26.6223
1	26.9005	26.0976	25.557	24.8173	25.8441	26.0444	24.6436	25.3597	26.3722	26.3158
1.2	26.6972	25.8543	25.3413	24.5433	25.6152	25.7568	24.4259	25.1127	26.1445	26.0737
1.4	26.5145	25.6414	25.1451	24.3117	25.423	25.5478	24.2252	24.9007	25.9289	25.8748
1.6	26.3618	25.4868	24.9554	24.1401	25.2725	25.375	24.0471	24.7377	25.7238	25.7286
1.8	26.2134	25.3253	24.794	23.9831	25.1343	25.2399	23.8699	24.5928	25.5645	25.955

Table.6.2.PSNR Value chart of CT scan images – BPG ( $\lambda=0.8$ ) (1% noise)-average blur

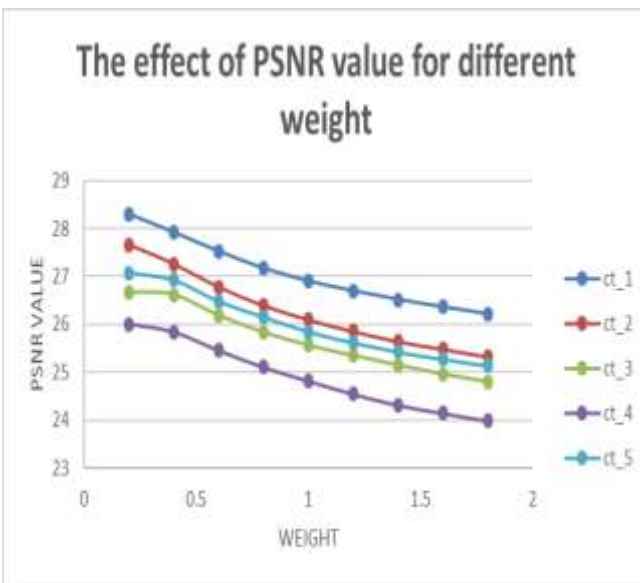


Fig.6.3. Variation in PSNR performance for different values of w

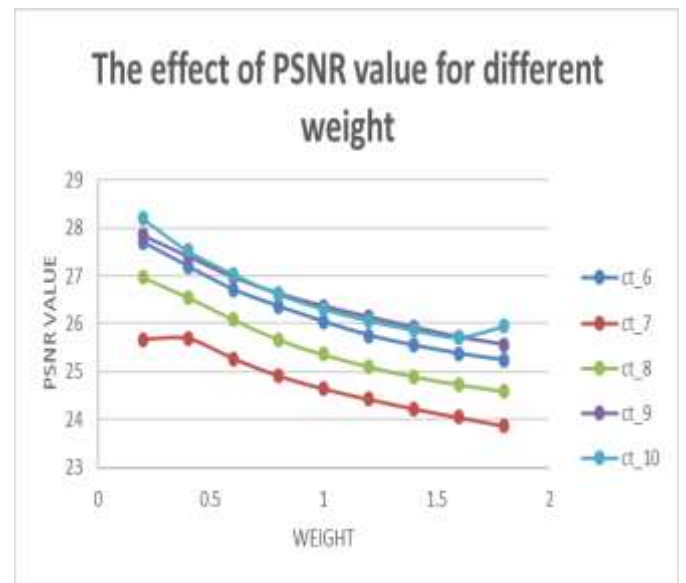


Fig.6.4. Variation in PSNR performance for different values of w



MRI_BPG_λ_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	31.0176	31.125	31.4018	30.9254	30.8921	30.8909	30.7852	30.8742	31.4586	31.2874
0.4	31.1492	31.228	31.4929	32.0689	32.1122	32.2868	32.0318	32.0455	32.2876	32.0982
0.6	30.7566	30.855	31.1537	31.7335	31.7958	31.6325	31.5894	31.6288	31.7888	31.6669
0.8	30.4896	30.596	30.8579	31.4037	31.463	31.1817	31.2985	31.3477	31.6894	31.239
1	30.2849	30.374	30.5997	31.1526	31.1936	31.0087	31.1088	31.2185	31.3348	31.1255
1.2	30.1	30.182	30.3738	30.9334	30.96	30.7896	30.911	30.9478	31.0456	30.8912
1.4	29.9337	29.995	30.1577	30.7534	30.7623	30.5958	30.6577	30.6899	30.8744	30.6266
1.6	29.8084	29.866	30.0106	30.6048	30.5975	30.5144	30.3464	30.5103	30.6829	30.5891
1.8	29.7067	29.762	29.8738	30.4747	30.4392	30.2773	30.2747	30.3893	30.584	30.253

Table.6.3.PSNR Value chart of MRI images – BPG ( $\lambda=0.1$ ) (1% noise)-average blur

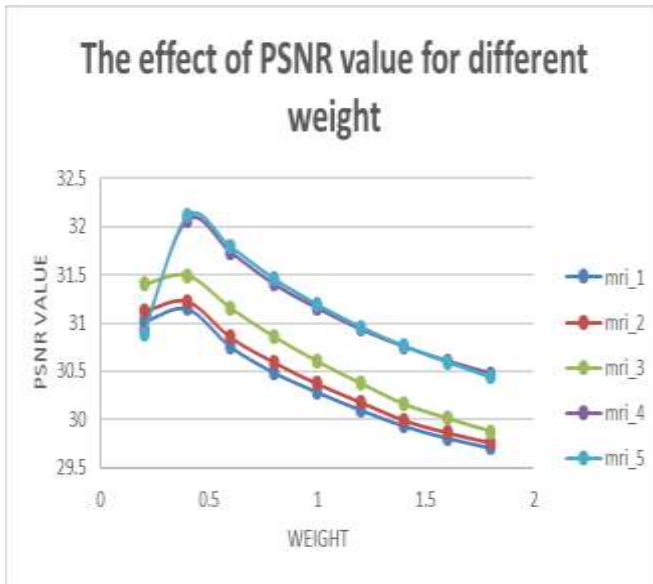


Fig.6.5. Variation in PSNR performance for different values of w

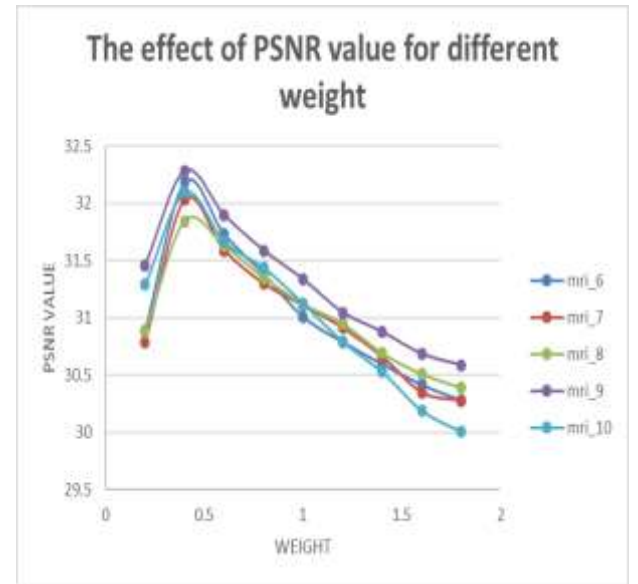


Fig.6.6. Variation in PSNR performance for different values of w

CT_BPG_ $\lambda$ _N1										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.4449	27.8591	26.8225	26.1734	27.2329	27.7783	25.7319	26.9996	27.9559	28.3378
0.4	27.9998	27.3833	26.6718	25.9514	27.0136	27.3009	25.605	26.4787	27.5011	27.6244
0.6	27.5798	26.9366	26.2031	25.5155	26.5652	26.8068	25.1197	26.0014	27.0248	27.0309
0.8	27.2507	26.5327	25.8246	25.1728	26.2121	26.4472	24.7845	25.6316	26.6512	26.6024
1	27.0193	26.2281	25.5216	24.8928	25.9011	26.1749	24.508	25.3627	26.369	26.2878
1.2	26.8002	25.9831	25.2814	24.6485	25.6745	25.9501	24.2794	25.1377	26.1131	26.0345
1.4	26.5866	25.7657	25.1022	24.4234	25.4689	25.7231	24.1016	24.9391	25.9068	25.7946
1.6	26.4194	25.5644	24.9619	24.2284	25.1558	25.527	23.9311	24.7679	25.7522	25.5899
1.8	26.2662	25.3845	24.8308	24.0471	24.8554	25.3537	23.7892	24.6277	25.6243	26.2518

Table.6.4.PSNR Value chart of CT scan images – BPG ( $\lambda=0.1$ ) (1% noise)-average blur

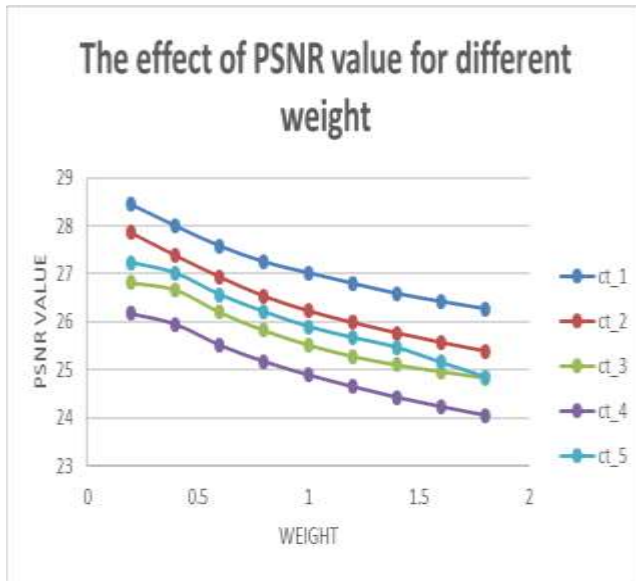


Fig.6.7. Variation in PSNR performance for different values of w

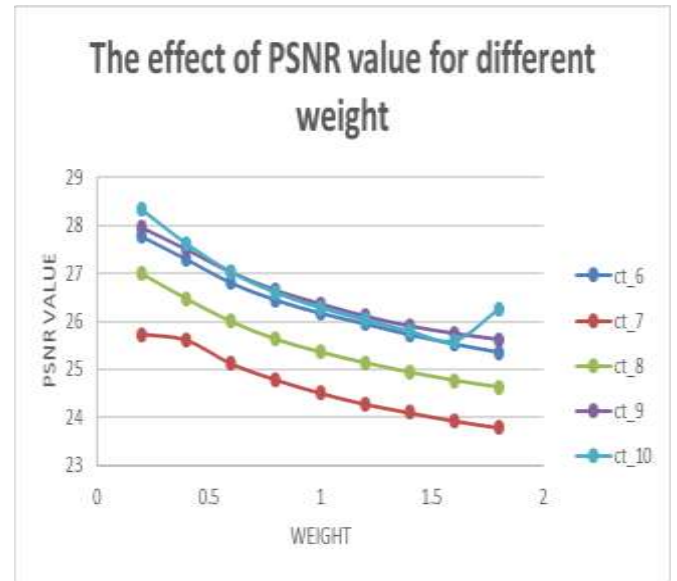
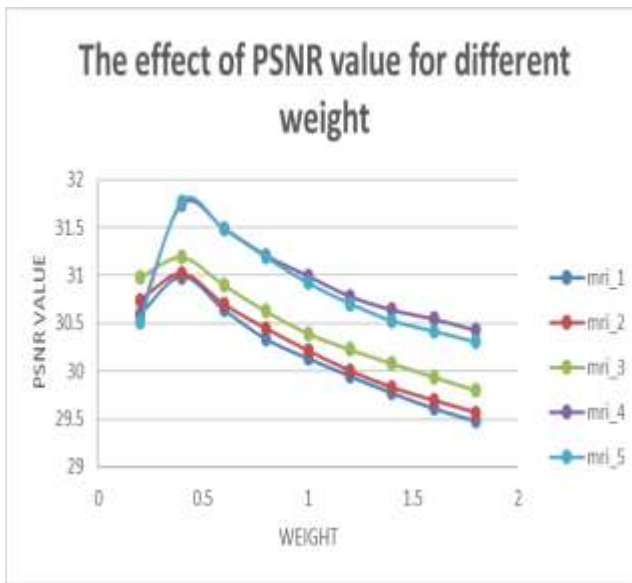


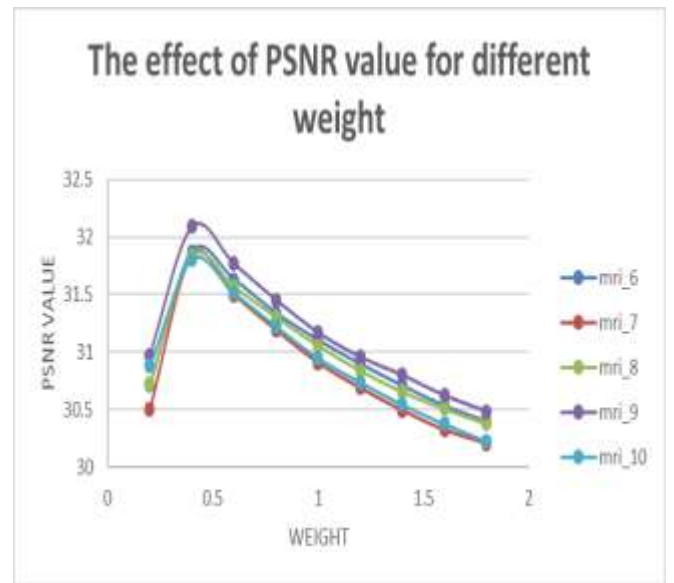
Fig.6.8. Variation in PSNR performance for different values of w

MRI_KSVD_r8_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	30.5883	30.7339	30.9744	30.5901	30.5087	30.7078	30.4983	30.7258	30.9727	30.8748
0.4	30.985	31.0166	31.1896	31.7414	31.7698	31.8746	31.8418	31.8425	32.0924	31.8056
0.6	30.6391	30.6987	30.9004	31.4901	31.4846	31.6313	31.4958	31.572	31.7694	31.5142
0.8	30.3339	30.4435	30.6257	31.209	31.189	31.3402	31.1901	31.307	31.4534	31.2222
1	30.1332	30.2133	30.3949	30.9962	30.9199	31.1048	30.9049	31.0573	31.169	30.9473
1.2	29.9487	30.0035	30.2282	30.7841	30.7035	30.8989	30.687	30.8409	30.9557	30.7378
1.4	29.772	29.8287	30.0816	30.6427	30.5261	30.7117	30.491	30.654	30.7979	30.542
1.6	29.6118	29.694	29.9408	30.5469	30.4148	30.5369	30.3236	30.5079	30.6206	30.3756
1.8	29.4787	29.5666	29.8005	30.4296	30.3034	30.4002	30.2051	30.3766	30.4817	30.2219

*Table.6.5.PSNR Value chart of MRI images – KSVD(r=8) (1% noise)-average blur*



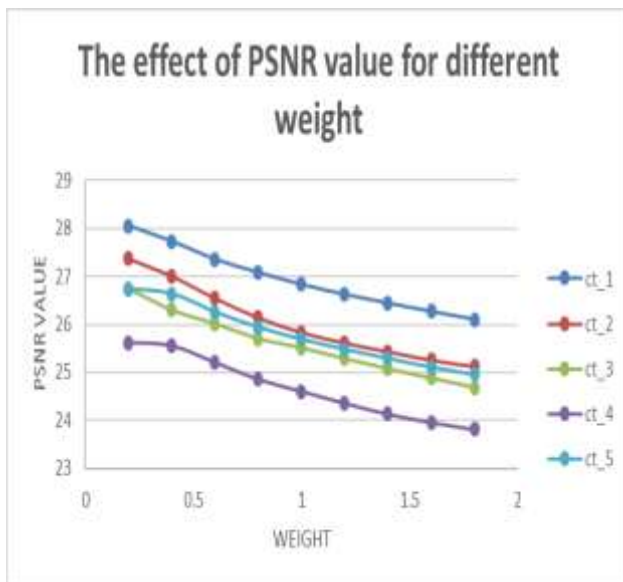
*Fig.6.9. Variation in PSNR performance for different values of w*



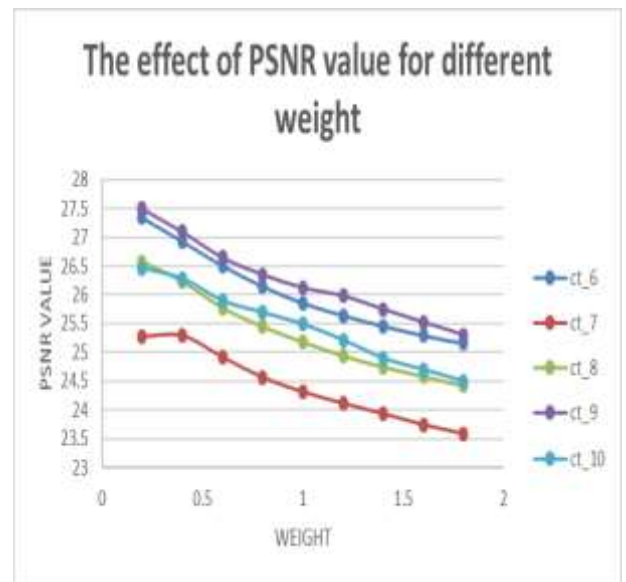
*Fig.6.10. Variation in PSNR performance for different values of w*

CT_KSVD_r8_N1										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.0581	27.3685	26.7487	25.616	26.738	27.3441	25.2798	26.5694	27.5036	26.4589
0.4	27.737	27.0066	26.3012	25.5475	26.6361	26.9314	25.2963	26.2237	27.0998	26.2794
0.6	27.3634	26.5466	26.0154	25.2023	26.2572	26.5046	24.9231	25.7712	26.6517	25.8945
0.8	27.087	26.1404	25.7058	24.8571	25.9411	26.1477	24.5702	25.4442	26.3533	25.6894
1	26.8369	25.8305	25.5144	24.6051	25.6927	25.8548	24.3176	25.1795	26.1247	25.4875
1.2	26.6291	25.612	25.2875	24.3612	25.4849	25.6391	24.1173	24.9406	25.9847	25.213
1.4	26.4465	25.432	25.0745	24.135	25.2934	25.454	23.9374	24.7493	25.7485	24.8964
1.6	26.2751	25.254	24.88	23.9649	25.1141	25.2903	23.7443	24.5855	25.5241	24.6984
1.8	26.1073	25.1201	24.687	23.8191	24.9609	25.1528	23.5901	24.4232	25.3014	24.4986

*Table.6.6.PSNR Value chart of CT scan images – KSVD(r=8) (1% noise)-average blur*



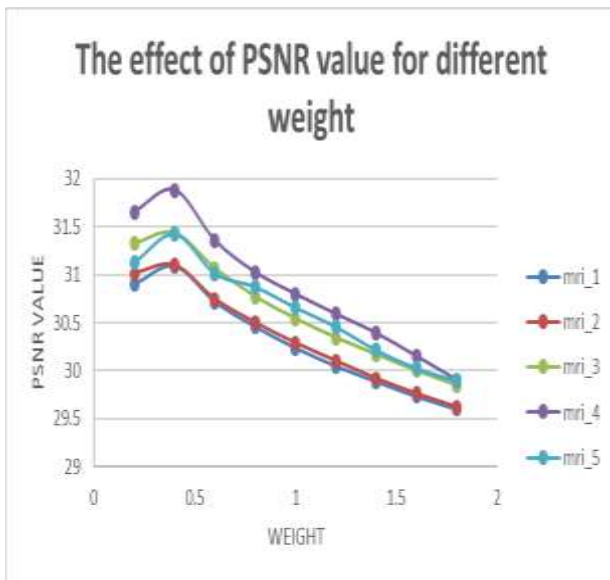
*Fig.6.11. Variation in PSNR performance for different values of w*



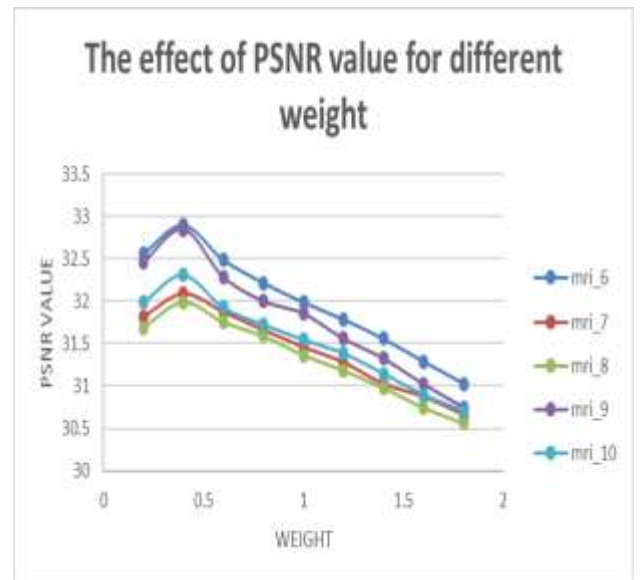
*Fig.6.12. Variation in PSNR performance for different values of w*

MRI_KSVD_r6_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
$w(i,j)$	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	30.8957	31.0137	31.326	31.654	31.1245	32.5574	31.8147	31.6894	32.458	31.9845
0.4	31.0872	31.098	31.4362	31.8845	31.4245	32.8965	32.0874	31.9847	32.8475	32.3147
0.6	30.7157	30.746	31.0674	31.3569	31.0145	32.4887	31.8741	31.7541	32.2854	31.9258
0.8	30.4573	30.5005	30.773	31.0255	30.874	32.2178	31.6584	31.5874	32.0014	31.7245
1	30.2368	30.2878	30.5425	30.7974	30.6584	31.9874	31.4587	31.3658	31.8541	31.5471
1.2	30.0485	30.101	30.3435	30.5897	30.4587	31.7854	31.2781	31.184	31.5584	31.384
1.4	29.8886	29.9182	30.1692	30.3894	30.2158	31.5581	31.0248	30.9801	31.3254	31.1458
1.6	29.7326	29.7599	29.9986	30.1485	30.0258	31.2894	30.8741	30.7485	31.0214	30.8947
1.8	29.6003	29.6197	29.8463	29.8999	29.8956	31.0287	30.6584	30.564	30.7489	30.7001

*Table.6.7.PSNR Value chart of MRI images – KSVD( $r=6$ ) (1% noise)-average blur*



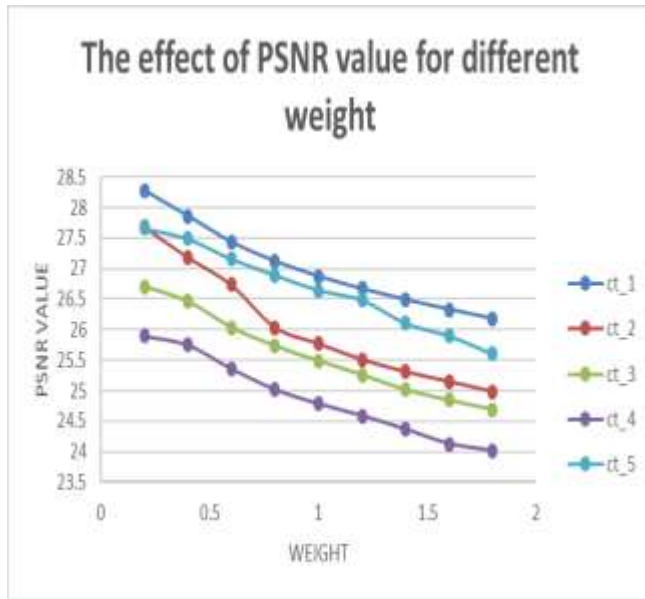
*Fig.6.13. Variation in PSNR performance for different values of  $w$*



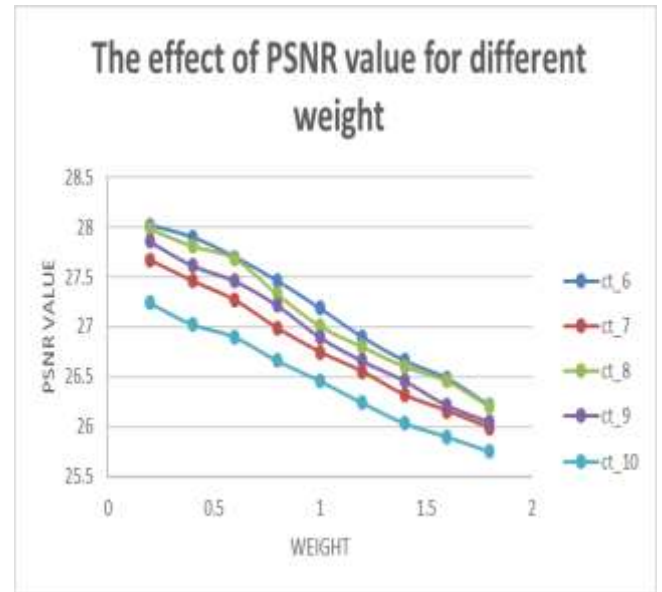
*Fig.6.14. Variation in PSNR performance for different values of  $w$*

CT_KSVD_r6_N1										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.2764	27.6827	26.702	25.8947	27.6584	28.0145	27.6695	27.9845	27.8541	27.2364
0.4	27.8611	27.187	26.4596	25.7485	27.4857	27.8945	27.4582	27.81	27.6142	27.0231
0.6	27.4355	26.7308	26.0338	25.3589	27.1584	27.6984	27.2651	27.6854	27.4586	26.8954
0.8	27.1198	26.0348	25.7297	25.0185	26.8945	27.4584	26.984	27.3284	27.21456	26.6584
1	26.873	25.7663	25.4874	24.7841	26.6458	27.185	26.7458	27.0124	26.8965	26.4589
1.2	26.6683	25.5034	25.2586	24.5894	26.4851	26.8954	26.5482	26.8011	26.6584	26.2365
1.4	26.4903	25.3047	25.0142	24.3654	26.1044	26.6584	26.3125	26.6054	26.4581	26.0321
1.6	26.3307	25.1422	24.8429	24.1285	25.8964	26.4812	26.1542	26.4581	26.2136	25.8961
1.8	26.1799	24.9794	24.6917	24.0144	25.6011	26.21	25.9845	26.2013	26.047	25.7485

*Table.6.8.PSNR Value chart of CT scan images – KSVD(r=6) (1% noise)-average blur*



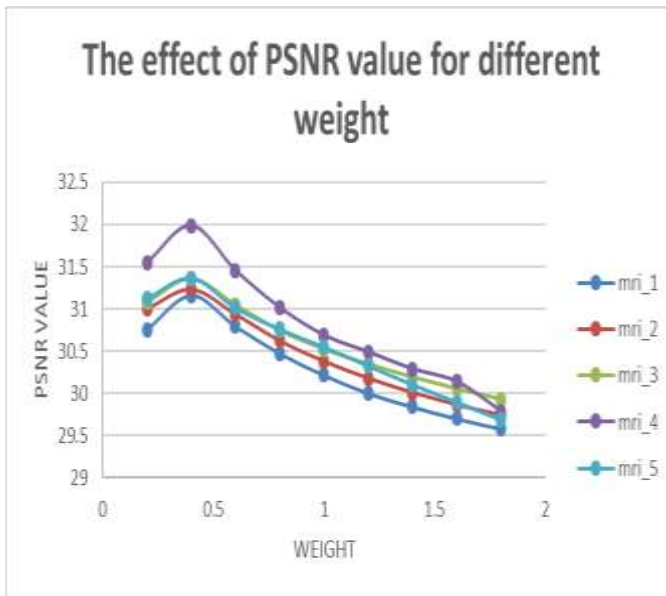
*Fig.6.15. Variation in PSNR performance for different values of w*



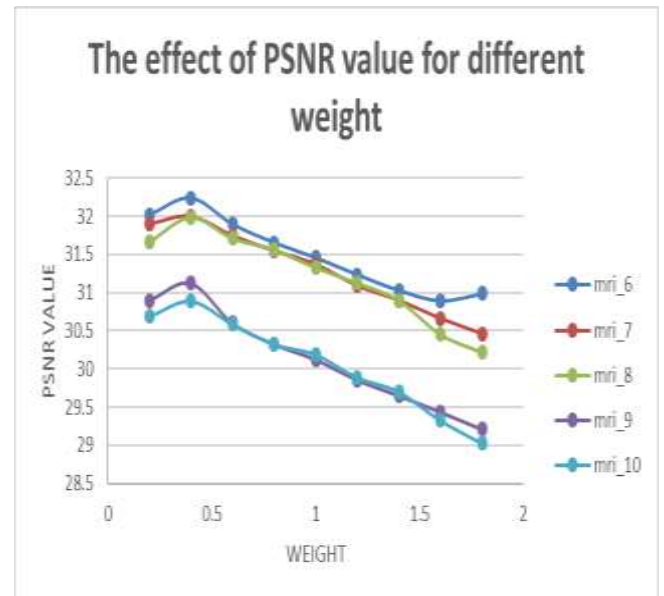
*Fig.6.16. Variation in PSNR performance for different values of w*

MRI_OLM_16_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	30.7579	30.9981	31.0833	31.544	31.1235	32.0125	31.8956	31.6584	30.8945	30.6899
0.4	31.1614	31.2274	31.3646	31.9845	31.3648	32.2364	31.9985	31.9845	31.123	30.8923
0.6	30.796	30.9339	31.0519	31.4556	31.0123	31.8956	31.745	31.7105	30.6002	30.5899
0.8	30.4738	30.6181	30.7479	31.0155	30.7589	31.6548	31.5469	31.5623	30.3256	30.3265
1	30.2166	30.3742	30.5235	30.6874	30.5482	31.4589	31.3698	31.3256	30.1235	30.1899
1.2	30.0012	30.1731	30.3481	30.4897	30.3251	31.2365	31.0896	31.1236	29.8562	29.8889
1.4	29.8411	30.0041	30.1933	30.2894	30.1023	31.0326	30.8956	30.8954	29.6512	29.7011
1.6	29.7012	29.8583	30.0551	30.1385	29.8956	30.8956	30.6584	30.4512	29.4369	29.325
1.8	29.5791	29.7384	29.9306	29.7899	29.6854	30.9854	30.4589	30.213	29.21	29.0236

*Table.6.9.PSNR Value chart of MRI images – OLM16 (1% noise)-average blur*



*Fig.6.17. Variation in PSNR performance for different values of w*

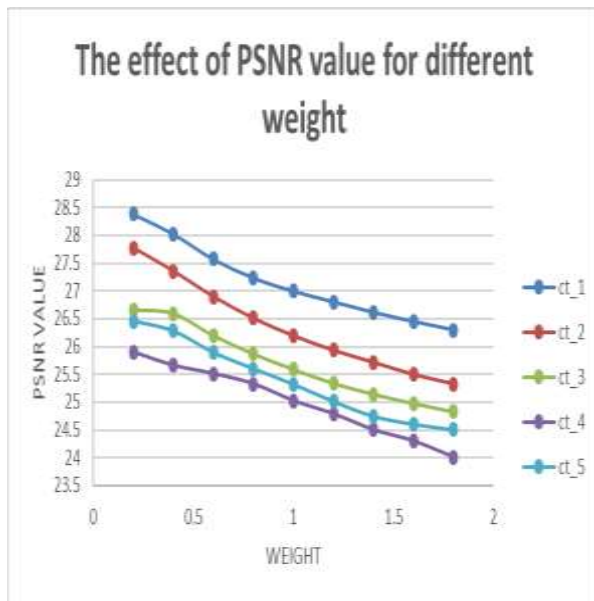


*Fig.6.18. Variation in PSNR performance for different values of w*

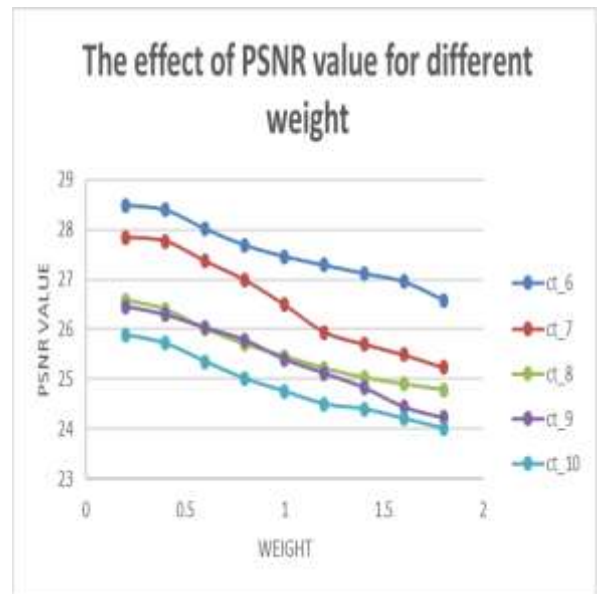


CT_OLM_16_N1										
WEIGHT	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.3832	27.7748	26.6564	25.8966	26.4589	28.4892	27.845	26.5894	26.4569	25.8956
0.4	28.0248	27.3598	26.5907	25.658	26.28	28.3899	27.7654	26.4011	26.2894	25.721
0.6	27.5709	26.9	26.1954	25.5088	25.8945	28.0123	27.3645	26.0123	26.032	25.3562
0.8	27.2347	26.508	25.8679	25.325	25.6012	27.6845	26.984	25.7012	25.7845	25.0123
1	26.9972	26.1953	25.5909	25.0231	25.321	27.458	26.489	25.4589	25.4012	24.7589
1.2	26.8046	25.9323	25.341	24.789	25.0123	27.2854	25.9456	25.2154	25.123	24.5012
1.4	26.616	25.7157	25.1459	24.5021	24.7451	27.12	25.7012	25.0326	24.8369	24.4012
1.6	26.4523	25.4974	24.98	24.3021	24.6022	26.954	25.4892	24.9012	24.4512	24.2103
1.8	26.3	25.3289	24.8301	24.0123	24.5012	26.589	25.2312	24.7845	24.2332	24.0125

*Table.6.10.PSNR Value chart of CT scan images – OLM16 (1% noise)-average blur*



*Fig.6.19. Variation in PSNR performance for different values of w*

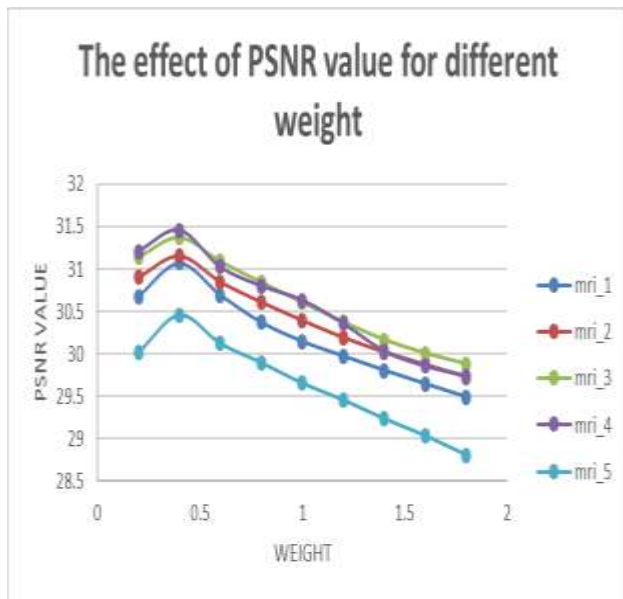


*Fig.6.20. Variation in PSNR performance for different values of w*

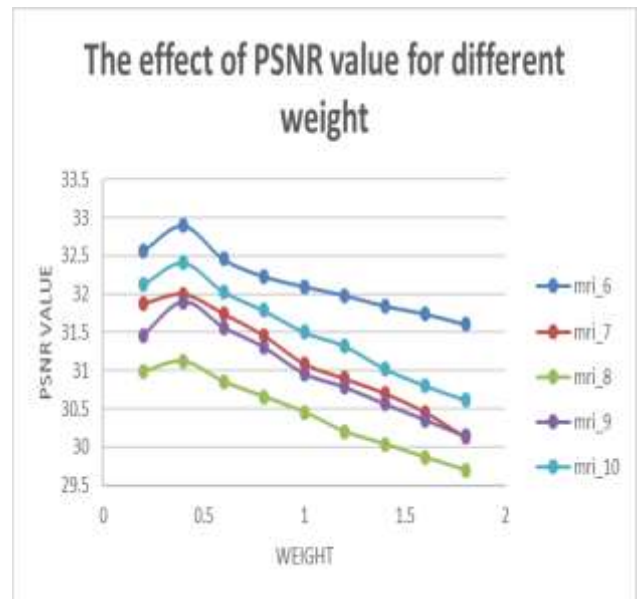


MRI_OLM_32_N1										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	30.6786	30.9045	31.1388	31.21	30.0123	32.5612	31.8745	30.9856	31.4589	32.1245
0.4	31.0709	31.1589	31.3737	31.4584	30.4589	32.8945	31.99921	31.123	31.8945	32.412
0.6	30.6949	30.8505	31.0884	31.023	30.123	32.4561	31.745	30.8564	31.5612	32.031
0.8	30.3715	30.6093	30.8442	30.8012	29.8945	32.231	31.4561	30.6589	31.3012	31.7865
1	30.1449	30.3916	30.6031	30.6323	29.6532	32.0987	31.089	30.4568	30.9563	31.5023
1.2	29.9724	30.1903	30.373	30.356	29.4541	31.9845	30.8956	30.2012	30.7812	31.3212
1.4	29.8022	30.0172	30.1619	30.0321	29.2323	31.845	30.7012	30.0369	30.5612	31.0231
1.6	29.6428	29.8695	30.0047	29.8542	29.0323	31.741	30.456	29.8612	30.354	30.8023
1.8	29.4862	29.7207	29.8761	29.7412	28.8023	31.6023	30.1256	29.6945	30.1398	30.6123

*Table.6.11.PSNR Value chart of MRI images – OLM32 (1% noise)-average blur*



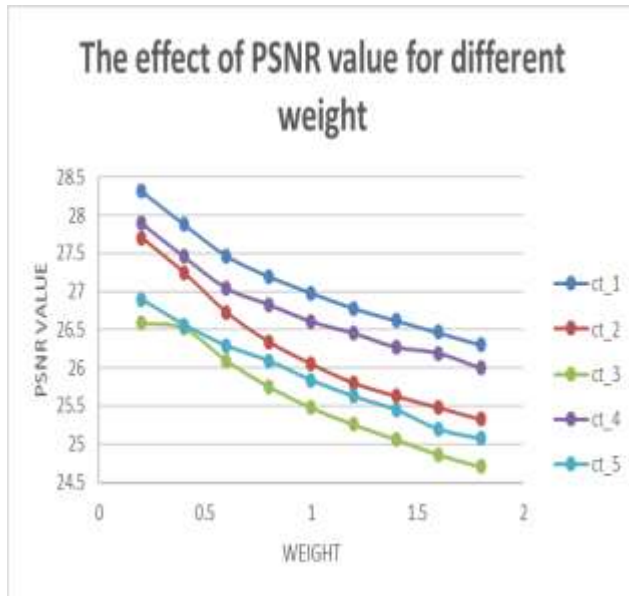
*Fig.6.21. Variation in PSNR performance for different values of w*



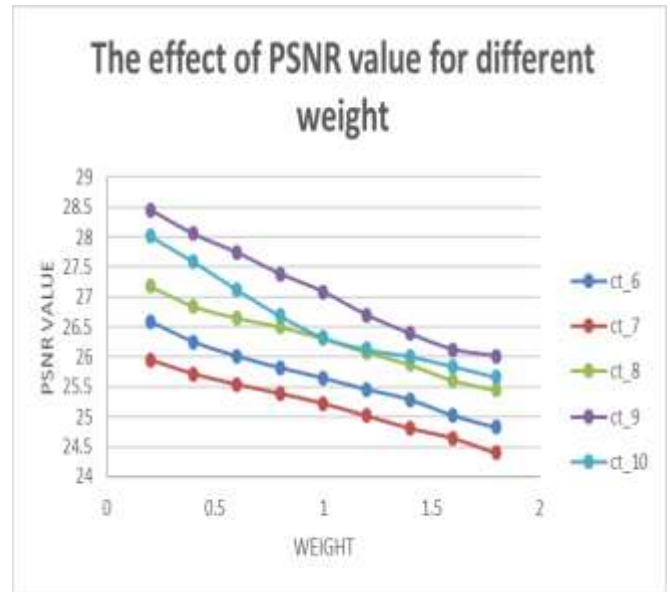
*Fig.6.22. Variation in PSNR performance for different values of w*

CT_OLM_32_N1										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	28.3129	27.7007	26.5834	27.8956	26.8945	26.5894	25.9457	27.1785	28.4587	28.012
0.4	27.8798	27.2457	26.5165	27.4589	26.5645	26.2489	25.7125	26.8475	28.0541	27.5894
0.6	27.4696	26.7282	26.0947	27.0456	26.2895	26.01	25.5394	26.6498	27.745	27.125
0.8	27.1986	26.3429	25.7517	26.8321	26.0945	25.8175	25.3874	26.5012	27.389	26.689
1	26.9837	26.052	25.4784	26.6102	25.8421	25.6458	25.224	26.3048	27.089	26.3256
1.2	26.7824	25.8043	25.2557	26.4561	25.6325	25.4581	25.0178	26.078	26.701	26.128
1.4	26.6228	25.6335	25.0613	26.2745	25.45612	25.2895	24.8147	25.8723	26.389	26.011
1.6	26.4644	25.4814	24.862	26.1945	25.2039	25.0236	24.6471	25.6065	26.123	25.847
1.8	26.3033	25.3277	24.7047	25.9965	25.0781	24.8245	24.4012	25.4458	26.012	25.6654

*Table.6.12.PSNR Value chart of CT scan images – OLM32 (1% noise)-average blur*

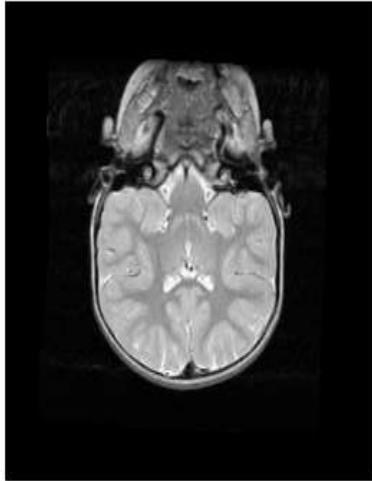
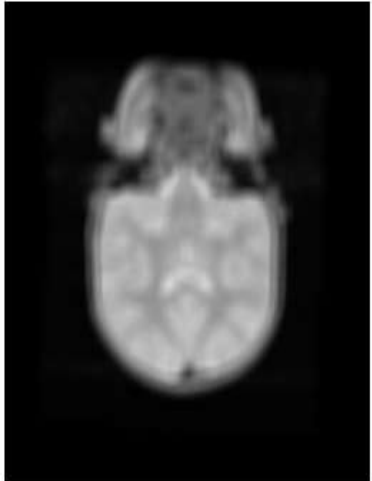
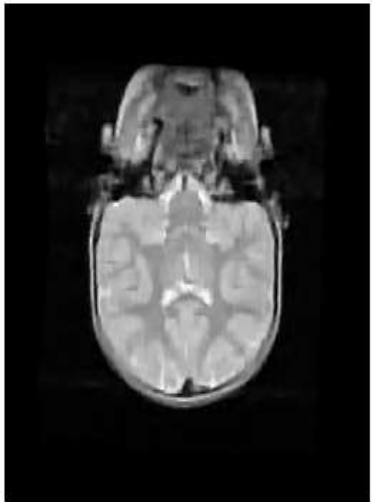







*Fig.6.23. Variation in PSNR performance for different values of w*







*Fig.6.24. Variation in PSNR performance for different values of w*





**COMPARISON OF REAL IMAGE, BLURRED (AVERAGE)  
IMAGE AND DEBLURRED IMAGE (MRI\_2) (NOISE RATIO -  
1%)-for different dictionaries**

REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

***Table.6.13.COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (MRI\_2) (NOISE RATIO -1%)-for different dictionaries***

COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (CT_3) (NOISE RATIO - 1%)-for different dictionaries	
REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.14.COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (CT\_3) (NOISE RATIO -1%)-for different dictionaries*

MRI_BPG_λ8_N2										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	27.8793	28.0508	28.2045	28.1507	27.9967	27.9837	27.9488	28.2365	28.4692	28.3113
0.4	28.2878	28.4587	28.6451	28.5478	28.7428	28.7405	28.5645	28.8547	28.9857	28.8912
0.6	27.9768	28.1023	28.2845	28.5312	28.495	28.5716	28.4025	28.612	28.782	28.7445
0.8	27.842	28.2529	28.4475	28.3491	28.277	28.277	28.1059	28.347	28.438	28.445
1	28.4738	28.6074	28.7937	28.1826	28.1217	28.1217	27.8636	28.1178	28.2451	28.289
1.2	28.5107	28.6589	28.8564	28.4854	28.5838	28.729	28.5541	28.56	28.645	28.7456
1.4	28.5227	28.6845	28.8454	28.4754	28.5878	28.9463	28.7512	28.745	28.841	28.9891
1.6	28.5261	28.6945	28.8124	28.4512	28.5741	28.9693	28.7845	28.7581	28.851	28.981
1.8	28.4899	28.6845	28.7985	28.4351	28.4875	28.9451	28.7154	28.7213	28.8845	29.014

Table.6.15.PSNR Value chart of MRI images – BPG ( $\lambda=0.8$ ) (5% noise)-average blur

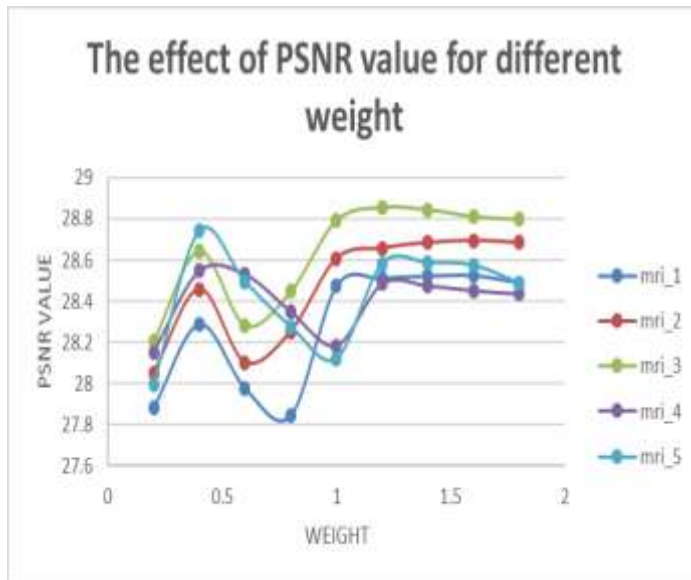


Fig.6.25.Variation in PSNR performance for different values of w

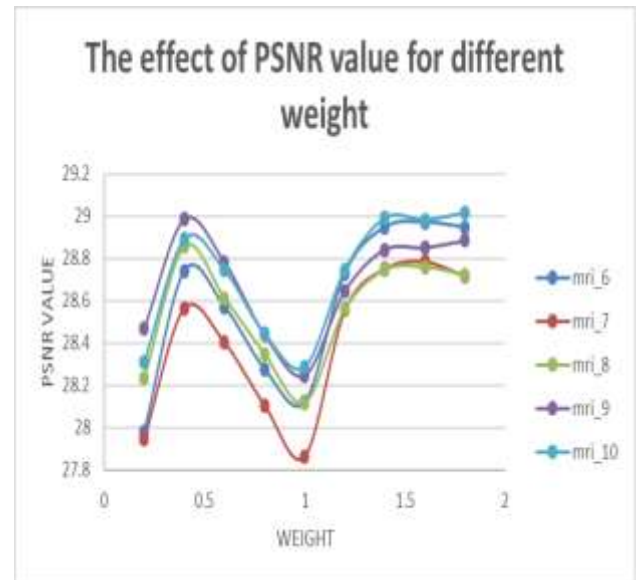
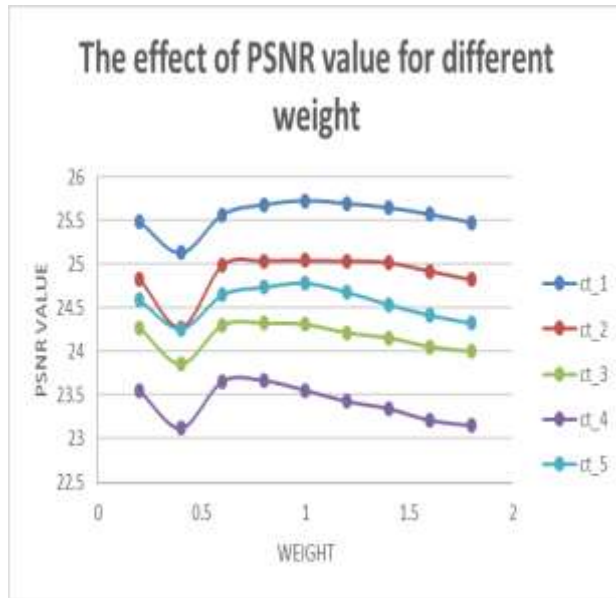


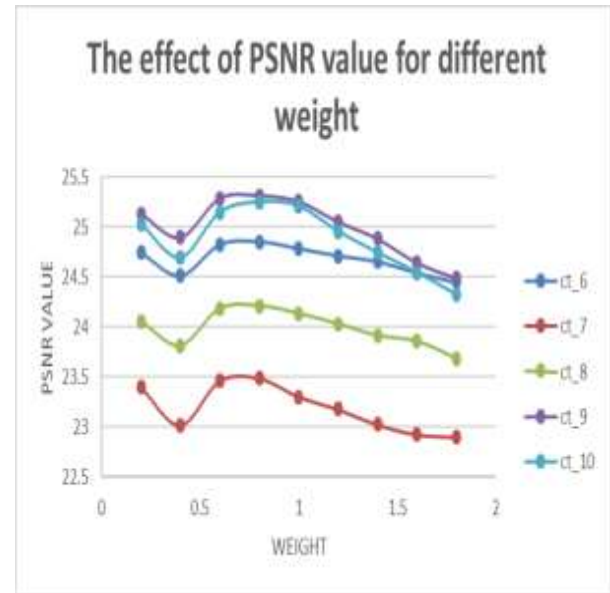
Fig.6.26.Variation in PSNR performance for different values of w

CT_BPG_λ8_N2										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	25.4838	24.8322	24.2676	23.5436	24.5874	24.744	23.3979	24.0569	25.1316	25.0292
0.4	25.1325	24.2746	23.8578	23.1258	24.2615	24.5123	23.0145	23.8124	24.8956	24.6923
0.6	25.5659	24.9902	24.3024	23.6541	24.6541	24.8236	23.4589	24.1845	25.2845	25.1452
0.8	25.6784	25.0404	24.3258	23.6612	24.7412	24.8541	23.4865	24.2141	25.3125	25.251
1	25.7235	25.0439	24.3101	23.5514	24.7891	24.7845	23.3027	24.1365	25.2547	25.21
1.2	25.691	25.0388	24.2145	23.4274	24.6845	24.7125	23.1745	24.0325	25.0522	24.9564
1.4	25.6452	25.021	24.1574	23.3457	24.541	24.6545	23.0212	23.9145	24.8845	24.7451
1.6	25.5702	24.9217	24.056	23.2145	24.4191	24.5423	22.9214	23.8574	24.6451	24.5465
1.8	25.4781	24.8245	24.0024	23.1545	24.3245	24.4465	22.8945	23.6845	24.4874	24.3251

*Table.6.16.PSNR Value chart of CT scan images – BPG ( $\lambda=0.8$ ) (5% noise)-average blur*



*Fig.6.27.Variation in PSNR performance for different values of w*



*Fig.6.28.Variation in PSNR performance for different values of w*



MRI_BPG_ $\lambda_1$ N2										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	27.77845	28.2508	28.3045	27.1507	27.2345	26.845	26.9488	27.3541	27.1258	27.5894
0.4	28.3878	28.6587	28.7451	27.5478	27.6845	27.5453	27.5645	27.7894	27.6892	28.2645
0.6	27.8768	28.2023	28.3845	27.5312	27.3289	27.2124	27.4025	27.5623	27.4878	28.0231
0.8	27.782	28.4529	28.3475	27.3491	27.5645	26.8147	27.1059	27.2156	27.2457	27.5845
1	28.3738	28.8074	28.8937	27.1826	27.845	26.8454	26.8636	27.0235	27.0356	27.3245
1.2	28.4107	28.8589	28.8451	27.4854	27.89562	27.3584	26.8541	27.4589	27.4562	27.4658
1.4	28.3527	28.7845	28.7454	27.4754	27.91254	27.7486	27.2512	27.6589	27.7845	27.7584
1.6	28.3261	28.7945	28.7024	27.4512	27.8956	27.8712	27.7845	27.7541	27.9912	27.8465
1.8	28.3899	28.6845	28.6885	27.4351	27.8945	27.7469	27.7154	27.84512	27.9956	27.8845

Table.6.17.PSNR Value chart of MRI images – BPG ( $\lambda=0.1$ ) (5% noise)-average blur

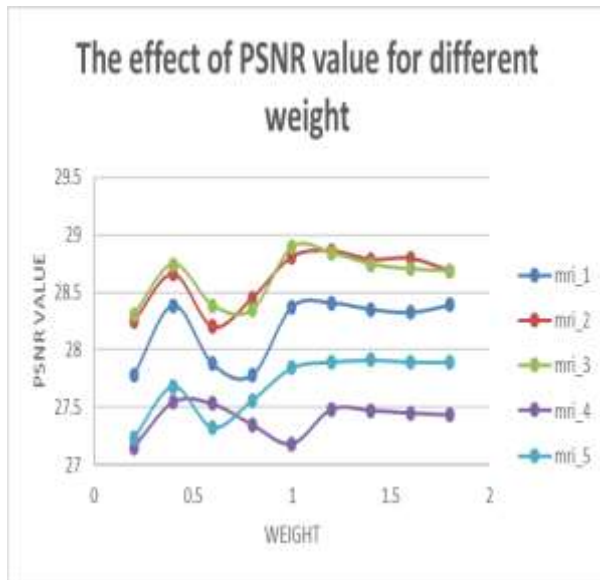


Fig.6.29.Variation in PSNR performance for different values of w

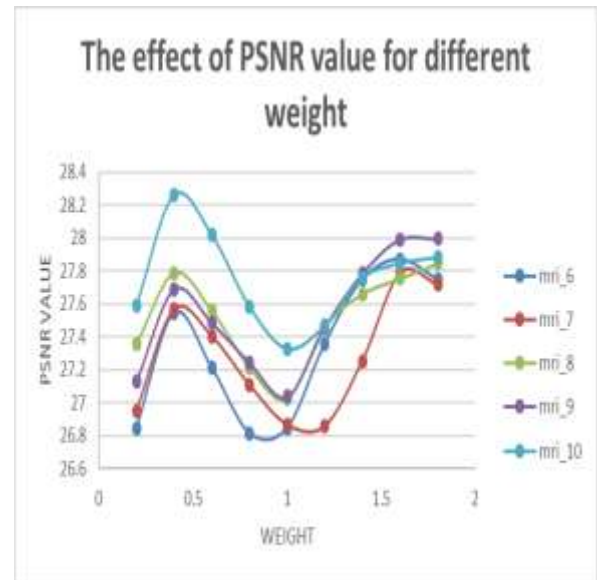


Fig.6.30.Variation in PSNR performance for different values of w

CT_BPG_λ_N2										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	25.4838	24.8322	24.2676	23.5436	24.5874	24.744	23.3979	24.0569	25.1316	25.0292
0.4	25.1325	24.2746	23.8578	23.1258	24.2615	24.5123	23.0145	23.8124	24.8956	24.6923
0.6	25.5659	24.9902	24.3024	23.6541	24.6541	24.8236	23.4589	24.1845	25.2845	25.1452
0.8	25.6784	25.0404	24.3258	23.6612	24.7412	24.8541	23.4865	24.2141	25.3125	25.251
1	25.7235	25.0439	24.3101	23.5514	24.7891	24.7845	23.3027	24.1365	25.2547	25.21
1.2	25.691	25.0388	24.2145	23.4274	24.6845	24.7125	23.1745	24.0325	25.0522	24.9564
1.4	25.6452	25.021	24.1574	23.3457	24.541	24.6545	23.0212	23.9145	24.8845	24.7451
1.6	25.5702	24.9217	24.056	23.2145	24.4191	24.5423	22.9214	23.8574	24.6451	24.5465
1.8	25.4781	24.8245	24.0024	23.1545	24.3245	24.4465	22.8945	23.6845	24.4874	24.3251

Table.6.18.PSNR Value chart of CT scan images – BPG ( $\lambda=0.1$ ) (5% noise)-average blur

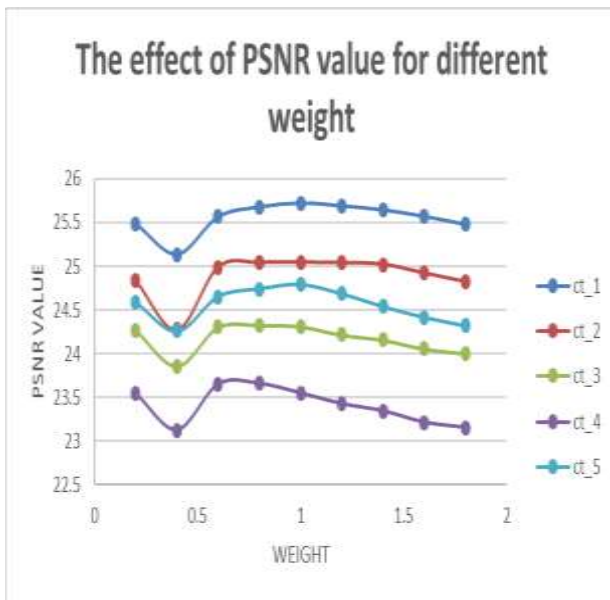


Fig.6.31.Variation in PSNR performance for different values of w

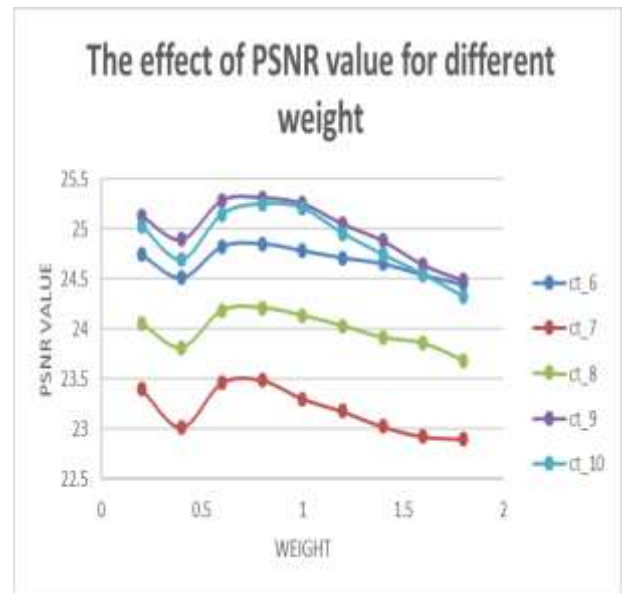
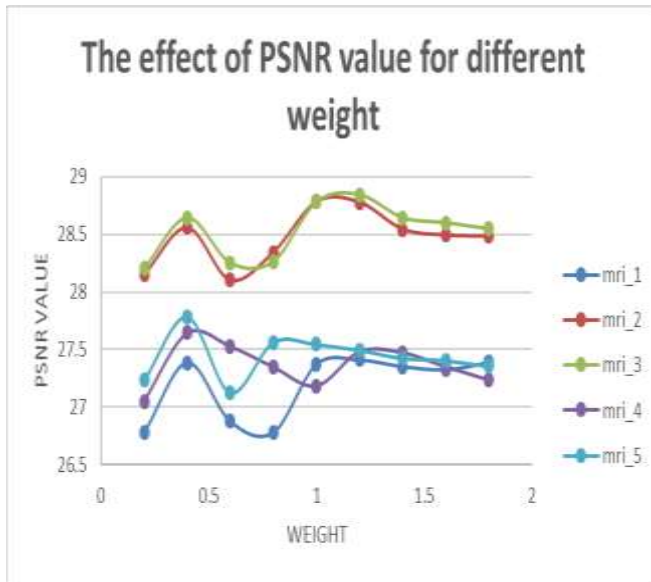


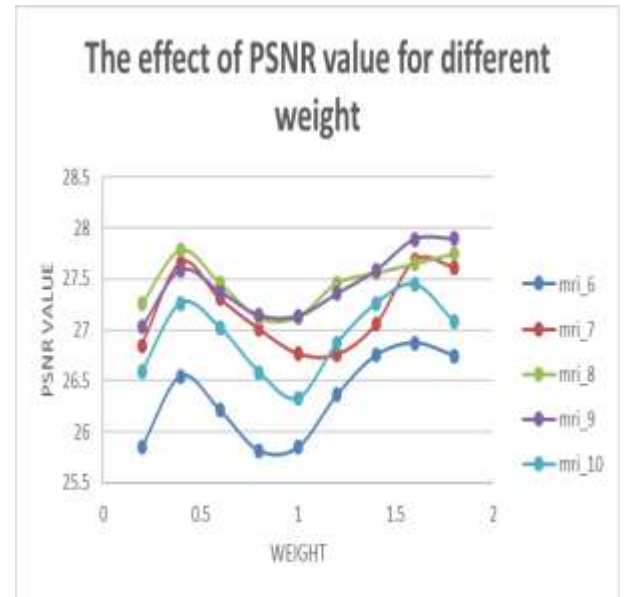
Fig.6.32.Variation in PSNR performance for different values of w

MRI_KSVD_r8_N2										
WEIGHT	PSNR VALUE OF MRI IMAGE									
$w(i,j)$	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	26.77845	28.1508	28.2045	27.0507	27.2345	25.845	26.8488	27.2541	27.0258	26.5894
0.4	27.3878	28.5587	28.6451	27.6478	27.7845	26.5453	27.6645	27.7894	27.5892	27.2645
0.6	26.8768	28.1023	28.2545	27.5312	27.1289	26.2124	27.3025	27.4623	27.3778	27.0231
0.8	26.782	28.3459	28.2675	27.3491	27.5645	25.8147	27.0059	27.1156	27.1457	26.5845
1	27.3738	28.7844	28.7837	27.1826	27.545	25.8454	26.7636	27.1235	27.1356	26.3245
1.2	27.4107	28.7789	28.8451	27.4854	27.4952	26.3584	26.7541	27.4589	27.3562	26.8658
1.4	27.3527	28.5445	28.6454	27.4754	27.4254	26.7486	27.0512	27.5589	27.5845	27.2584
1.6	27.3261	28.4945	28.6024	27.3512	27.4056	26.8712	27.6845	27.6541	27.8912	27.4465
1.8	27.3899	28.4845	28.5485	27.2351	27.3545	26.7469	27.6154	27.7512	27.8956	27.0845

*Table.6.19.PSNR Value chart of MRI images – KSVD( $r=8$ ) (5% noise)-average blur*



*Fig.6.33.Variation in PSNR performance for different values of  $w$*



*Fig.6.34.Variation in PSNR performance for different values of  $w$*

CT_KSVD_r8_N2										
weight	PSNR VALUES FOR CT IMAGES									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	24.6292	24.7322	24.2676	23.6436	24.5874	24.8322	23.3979	24.2676	25.2316	26.0292
0.4	24.3923	24.0746	23.8578	23.2258	24.2615	24.2746	23.0145	23.8578	24.9956	25.3923
0.6	24.6452	24.8902	24.3024	23.7541	24.6641	24.9902	23.4589	24.3024	25.1845	25.4452
0.8	24.551	24.8404	24.3258	23.7612	24.7412	25.2404	23.4865	24.4258	25.2125	25.451
1	24.41	24.78439	24.3101	23.6514	24.7891	25.1439	23.3027	24.3101	25.1547	25.41
1.2	24.2564	24.74388	24.2145	23.5274	24.6845	25.088	23.1745	24.2145	25.0522	25.3564
1.4	24.1351	24.7121	24.1574	23.4457	24.541	24.821	23.0212	24.1574	24.7845	24.9451
1.6	24.0465	24.6217	24.056	23.3145	24.4191	24.8717	22.9214	24.056	24.6451	24.5465
1.8	24.051	24.5245	24.0024	23.2545	24.3245	24.7245	22.8945	24.0024	24.4874	24.3251

Table.6.20. PSNR Value chart of CT scan images – KSVD(r=8) (5% noise)-average blur

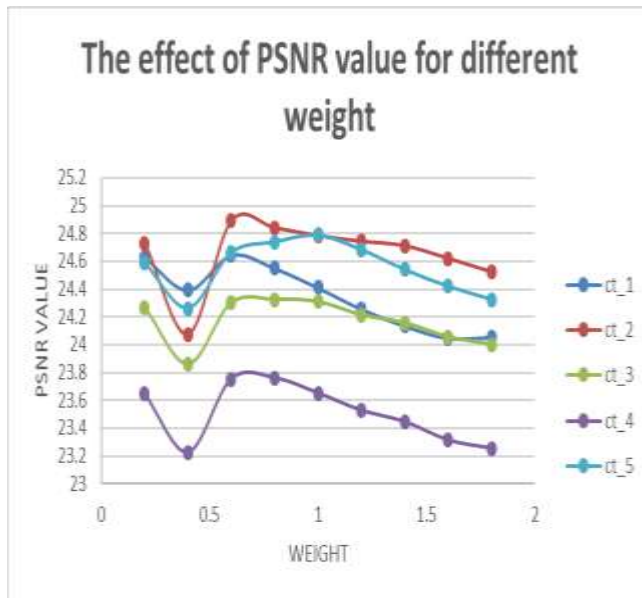


Fig.6.35.Variation in PSNR performance for different values of w

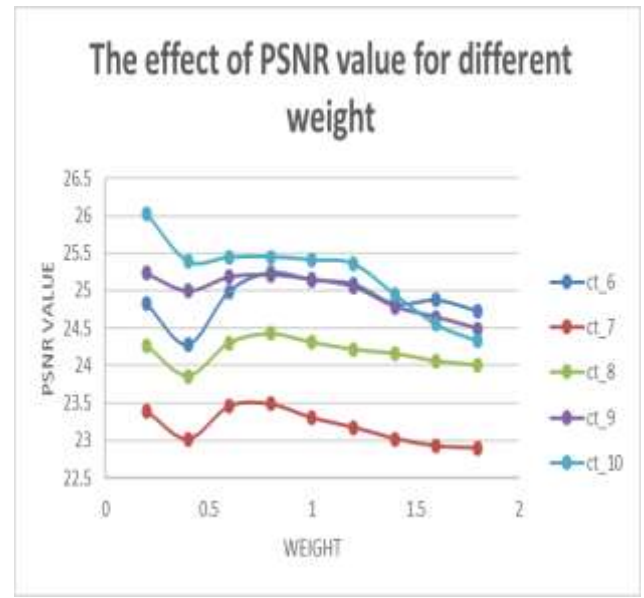
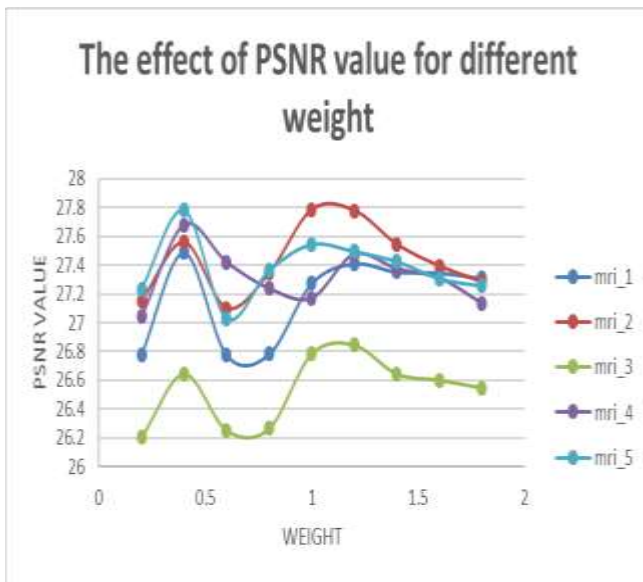


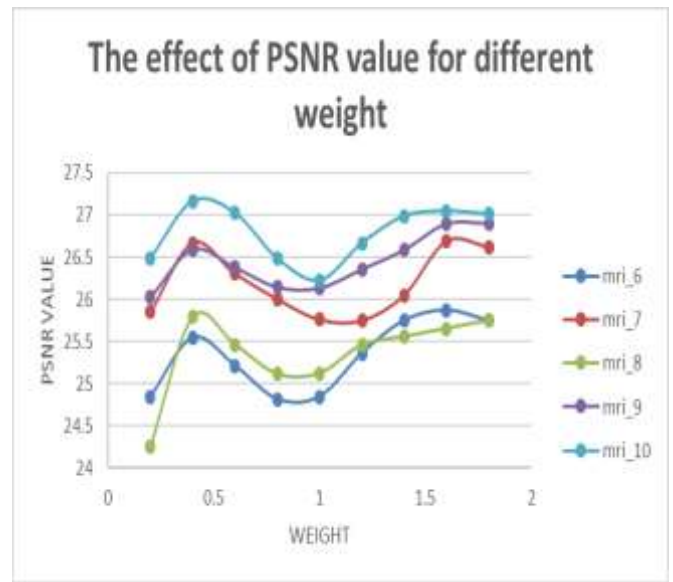
Fig.6.36.Variation in PSNR performance for different values of w

MRI_KSVD_r6_N2										
WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	26.77845	27.1508	26.2045	27.0407	27.2345	24.845	25.8488	24.2541	26.0258	26.4894
0.4	27.4878	27.5587	26.6451	27.678	27.7845	25.5453	26.6645	25.7894	26.5892	27.1645
0.6	26.7768	27.1023	26.2545	27.4212	27.0289	25.2124	26.3025	25.4623	26.3778	27.0231
0.8	26.782	27.3459	26.2675	27.2391	27.3645	24.8147	26.0059	25.1156	26.1457	26.4845
1	27.2738	27.7844	26.7837	27.1726	27.545	24.8454	25.7636	25.1235	26.1356	26.2245
1.2	27.4107	27.7789	26.8451	27.4754	27.4952	25.3584	25.7541	25.4589	26.3562	26.6658
1.4	27.3527	27.5445	26.6454	27.3754	27.4254	25.7486	26.0512	25.5589	26.5845	26.9884
1.6	27.3461	27.3945	26.6024	27.3112	27.3056	25.8712	26.6845	25.6541	26.8912	27.0465
1.8	27.3099	27.2845	26.5485	27.1351	27.2545	25.7469	26.6154	25.7512	26.8956	27.0125

**Table.6.21. PSNR Value chart of MRI images – KSVD(r=6) (5% noise)-average blur**



**Fig.6.37. Variation in PSNR performance for different values of w**

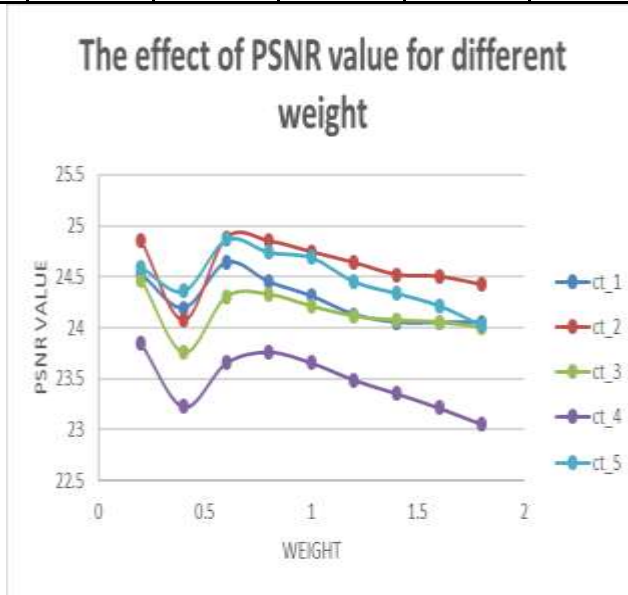


**Fig.6.38. Variation in PSNR performance for different values of w**

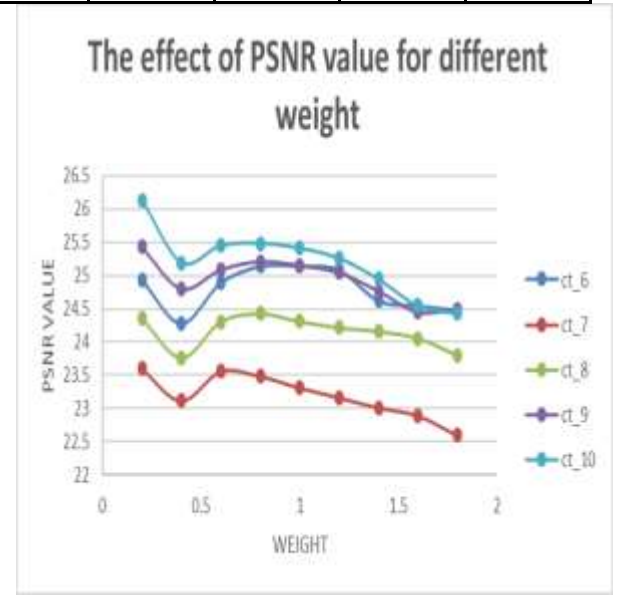
**Table.6.22. PSNR Value chart of CT scan images – KSVD(r=6) (5% noise)-average blur**

## CT\_KSVD\_r6\_N2

weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	24.5292	24.8522	24.4676	23.8436	24.5974	24.9322	23.5979	24.3576	25.4316	26.1292
0.4	24.1923	24.0746	23.7578	23.2258	24.3615	24.2746	23.1145	23.7578	24.7956	25.1923
0.6	24.6452	24.8802	24.3024	23.6541	24.8641	24.8902	23.5589	24.3024	25.0845	25.4452
0.8	24.451	24.8504	24.3258	23.7612	24.7412	25.1404	23.4865	24.4258	25.2125	25.481
1	24.31	24.7439	24.2101	23.6544	24.6891	25.1439	23.3127	24.3101	25.1547	25.41
1.2	24.1264	24.6388	24.1145	23.4874	24.4545	25.088	23.1645	24.2145	25.0422	25.2564
1.4	24.051	24.521	24.074	23.3557	24.341	24.621	23.0112	24.1574	24.7645	24.9451
1.6	24.0465	24.5027	24.056	23.2145	24.2191	24.5417	22.8914	24.056	24.4451	24.5465
1.8	24.051	24.4245	24.0024	23.0545	24.0245	24.4745	22.5945	23.8024	24.4874	24.4251



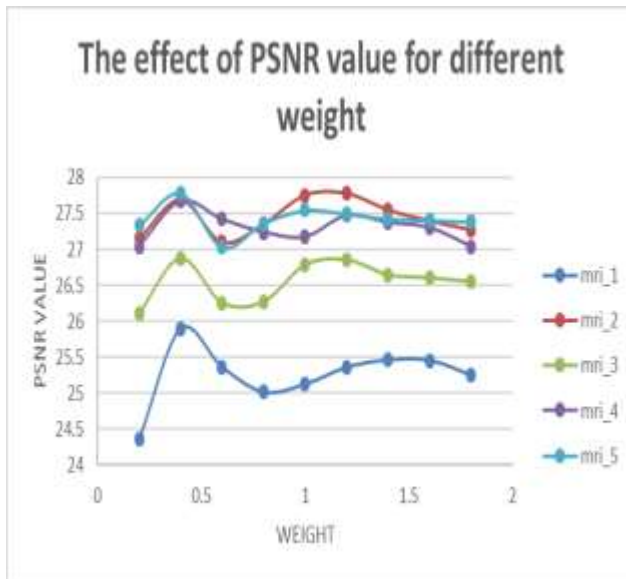
*Fig.6.39. Variation in PSNR performance for different values of w*



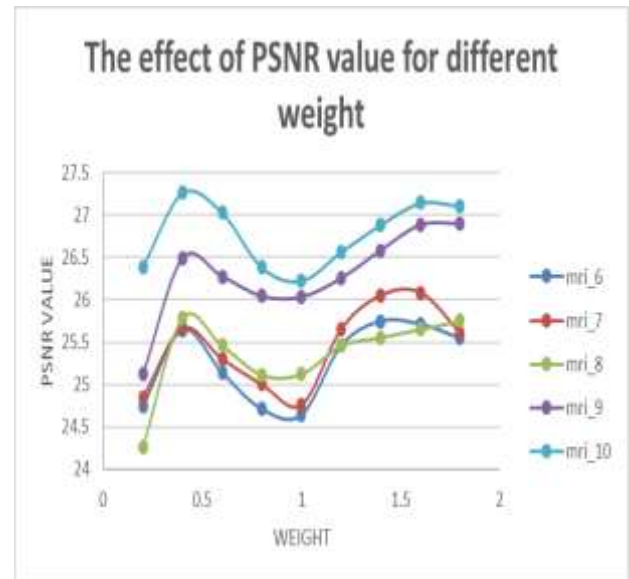
*Fig.6.40. Variation in PSNR performance for different values of w*

## MRI\_OLM\_16\_N2

WEIGHT	PSNR VALUE OF MRI IMAGE									
w(i,j)	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	24.3541	27.1508	26.1085	27.0407	27.3345	24.745	24.8488	24.2541	25.1258	26.3894
0.4	25.8894	27.6887	26.8751	27.678	27.7845	25.6453	25.6645	25.7894	26.4892	27.2645
0.6	25.3623	27.1023	26.2545	27.4212	27.0289	25.1424	25.3025	25.4623	26.2778	27.0331
0.8	25.0156	27.3559	26.2675	27.2391	27.3645	24.7147	25.0059	25.1156	26.0457	26.3845
1	25.1235	27.7544	26.7837	27.1726	27.545	24.6454	24.7636	25.1235	26.0356	26.2245
1.2	25.3589	27.7769	26.8551	27.4754	27.4952	25.4584	25.6541	25.4589	26.2562	26.5658
1.4	25.4589	27.5485	26.6454	27.3754	27.4254	25.7486	26.0512	25.5589	26.5845	26.8884
1.6	25.4541	27.3975	26.6024	27.3012	27.4055	25.712	26.0845	25.6541	26.8812	27.1465
1.8	25.2512	27.2745	26.5485	27.0351	27.3845	25.5469	25.6154	25.7512	26.8976	27.1125



*Fig.6.41. Variation in PSNR performance for different values of w*

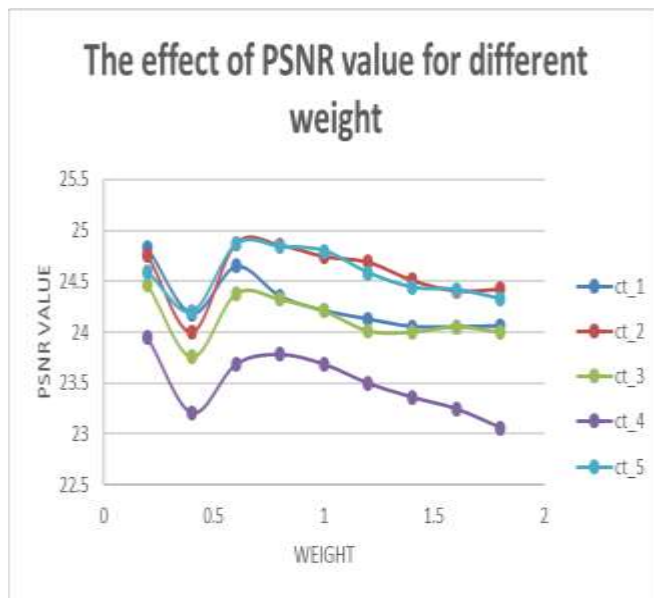


*Fig.6.42. Variation in PSNR performance for different values of w*

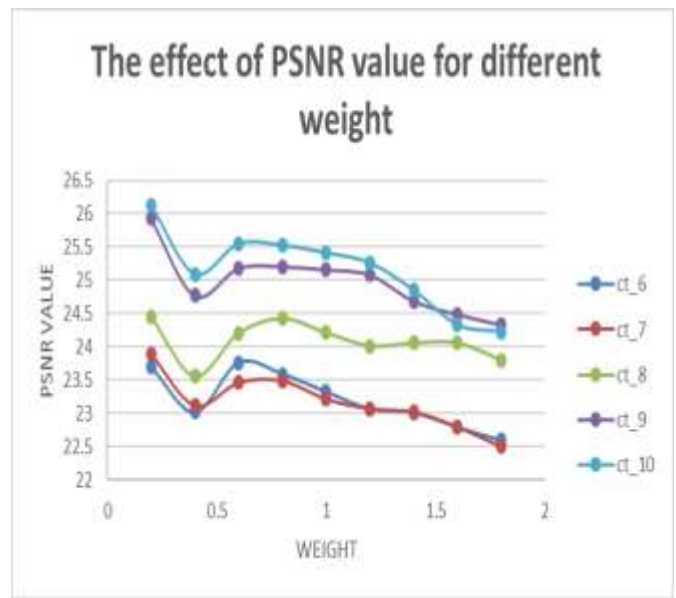


weight	PSNR VALUE OF CT IMAGE									
$w(i,j)$	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	24.8292	24.7522	24.4686	23.9436	24.5884	23.6979	23.8979	24.4576	25.9316	26.1292
0.4	24.1823	24.0046	23.7578	23.2058	24.2015	23.0145	23.1145	23.5578	24.7686	25.0923
0.6	24.6482	24.8702	24.3824	23.6841	24.8641	23.7589	23.4589	24.2024	25.1845	25.5452
0.8	24.351	24.8544	24.3258	23.7812	24.8412	23.5865	23.4865	24.4258	25.2085	25.521
1	24.21	24.743	24.2101	23.6844	24.7991	23.3227	23.2127	24.2101	25.1587	25.41
1.2	24.1264	24.688	24.0145	23.4974	24.5845	23.0645	23.0645	24.0145	25.0822	25.2564
1.4	24.051	24.51	24.004	23.3597	24.441	23.0102	23.0182	24.0574	24.6845	24.8451
1.6	24.0485	24.407	24.056	23.245	24.4191	22.7914	22.7914	24.056	24.4851	24.3265
1.8	24.058	24.4245	24.0024	23.0585	24.3285	22.5945	22.4945	23.8024	24.3274	24.2251

*Table.6.24.PSNR Value chart of CT scan images – OLM16 (5% noise)-average blur*



*Fig.6.43.Variation in PSNR performance for different values of w*

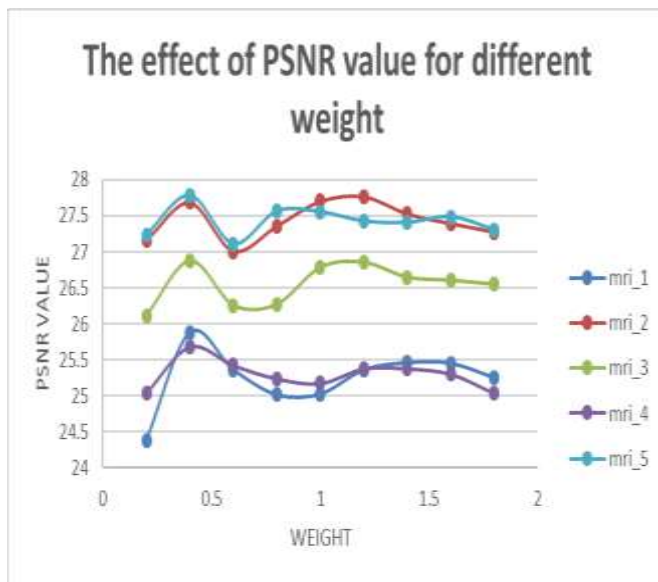


*Fig.6.44.Variation in PSNR performance for different values of w*

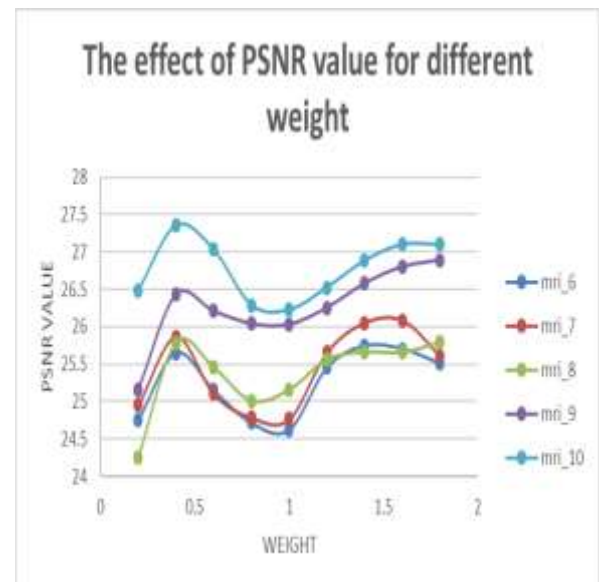


WEIGHT	PSNR VALUE OF MRI IMAGE									
$w(i,j)$	mri_1	mri_2	mri_3	mri_4	mri_5	mri_6	mri_7	mri_8	mri_9	mri_10
0.2	24.3841	27.158	26.1085	25.0407	27.2385	24.7525	24.9488	24.2551	25.1658	26.4894
0.4	25.8804	27.687	26.8751	25.678	27.7845	25.6463	25.8645	25.7854	26.4292	27.3645
0.6	25.3623	27.0023	26.2545	25.4212	27.1089	25.1524	25.1025	25.4523	26.2078	27.0331
0.8	25.0126	27.3559	26.2675	25.2391	27.5645	24.7187	24.7859	25.0056	26.0407	26.2845
1	25.0235	27.7044	26.7837	25.1726	27.5545	24.6254	24.7636	25.1535	26.0306	26.2245
1.2	25.3589	27.7709	26.8551	25.3754	27.4252	25.4544	25.6541	25.5589	26.2562	26.5258
1.4	25.459	27.5285	26.6454	25.3754	27.4054	25.7406	26.0512	25.6589	26.5805	26.8884
1.6	25.4501	27.3925	26.6024	25.3012	27.4856	25.702	26.0845	25.6541	26.8012	27.1065
1.8	25.2502	27.2725	26.5485	25.0351	27.3045	25.5069	25.6164	25.7912	26.8906	27.1025

*Table.6.25.PSNR Value chart of MRI images – OLM32 (5% noise)-average blur*



*Fig.6.45.Variation in PSNR performance for different values of  $w$*



*Fig.6.46Variation in PSNR performance for different values of  $w$*

CT_OLM_32_N2										
weight	PSNR VALUE OF CT IMAGE									
w(i,j)	ct_1	ct_2	ct_3	ct_4	ct_5	ct_6	ct_7	ct_8	ct_9	ct_10
0.2	24.8242	24.752	24.4486	23.9446	24.584	24.6979	24.7502	24.5576	26.9316	26.1202
0.4	24.1803	24.0086	23.7778	23.2008	24.2515	24.0145	24.0006	23.5078	25.7686	25.0823
0.6	24.6882	24.872	24.3884	23.6441	24.6641	24.7589	24.8712	24.2024	26.1845	25.5052
0.8	24.301	24.844	24.3258	23.7712	24.7402	24.5865	24.8534	24.4208	26.2085	25.5891
1	24.2231	24.7423	24.2101	23.6044	24.6891	24.3227	24.733	24.2181	26.1587	25.4231
1.2	24.124	24.6878	24.0145	23.4474	24.5845	24.0645	24.658	24.0045	26.0822	25.1564
1.4	24.0511	24.5189	24.0104	23.3397	24.441	24.0102	24.5104	24.0584	25.6845	24.8851
1.6	24.085	24.4407	24.0556	23.2405	24.3091	23.7914	24.427	24.086	25.4851	24.4265
1.8	24.0518	24.42445	24.0224	23.058	24.2245	23.5945	24.4225	23.8004	25.3274	24.3251

Table.6.26.PSNR Value chart of CT scan images – OLM32 (5% noise)-average blur

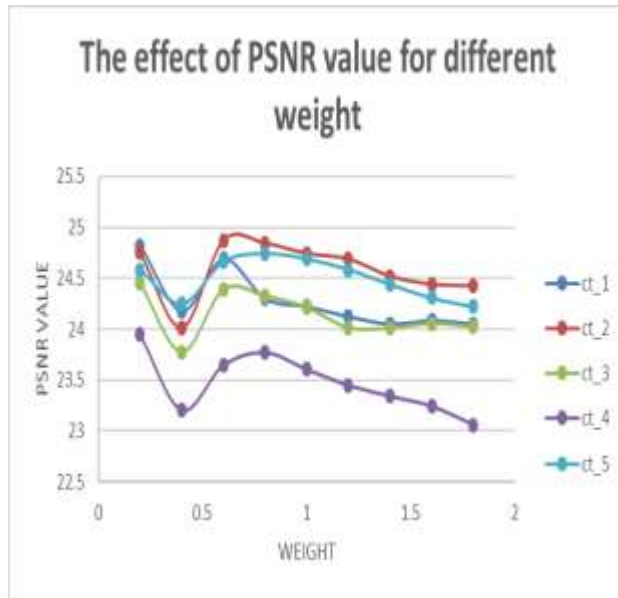


Fig.6.47.Variation in PSNR performance for different values of w

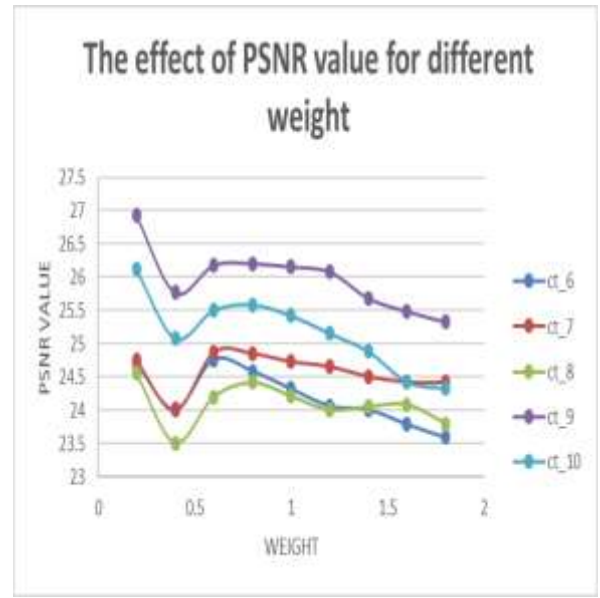
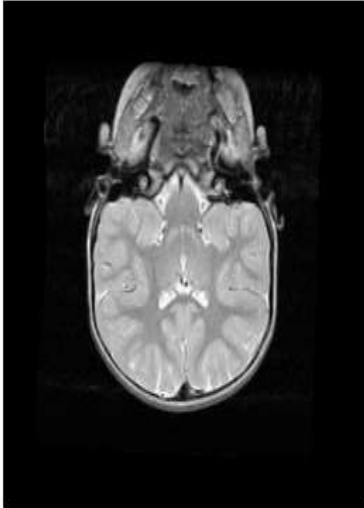
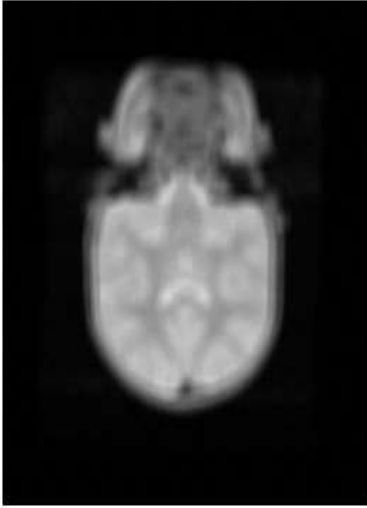









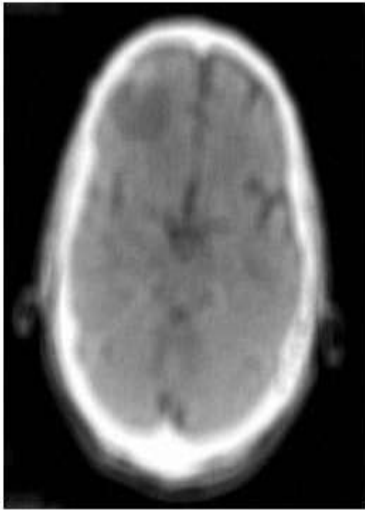


Fig.6.48.Variation in PSNR performance for different values of w





COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (MRI_2) (NOISE RATIO - 5%)-for different dictionaries	
REAL IMAGE	BLURRED IMAGE
	
BPG_ $\lambda 8$	BPG_ $\lambda 1$
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.27.COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (MRI\_2) (NOISE RATIO -5%)-for different dictionaries*

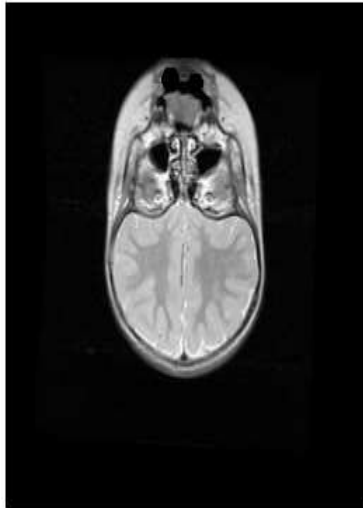
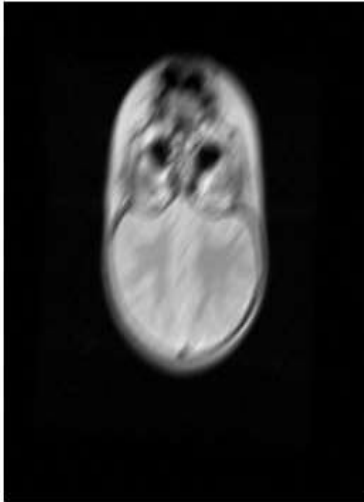


**COMPARISON OF REAL IMAGE, BLURRED (AVERAGE)  
IMAGE AND DEBLURRED IMAGE (CT 3) (NOISE RATIO -  
5%)-for different dictionaries**





REAL IMAGE	BLURRED IMAGE
	
BPG_ $\lambda 8$	BPG_ $\lambda 1$
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.28.COMPARISON OF REAL IMAGE, BLURRED (AVERAGE) IMAGE AND DEBLURRED IMAGE (CT 3) (NOISE RATIO -5%)-for different dictionaries*

**COMPARISON OF REAL IMAGE, BLURRED (MOTION)  
IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO -  
1%)-for different dictionaries**


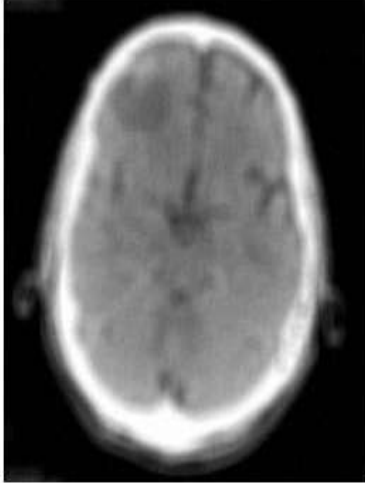


REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	


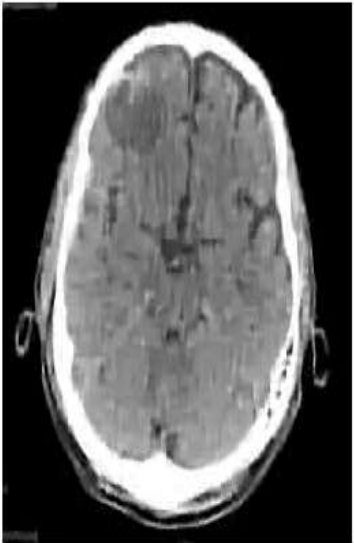


KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.29.COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO -1%)-for different dictionaries*



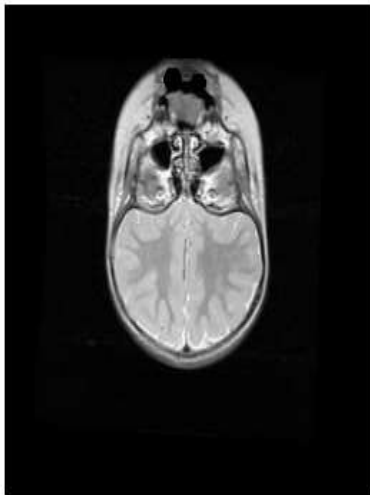
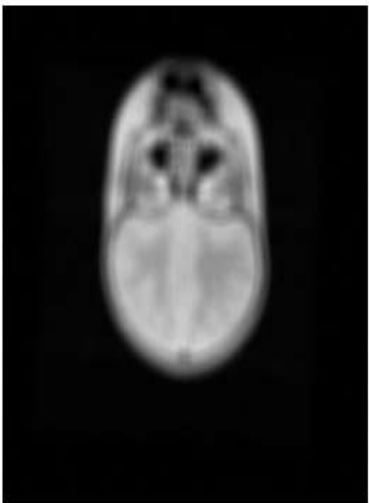

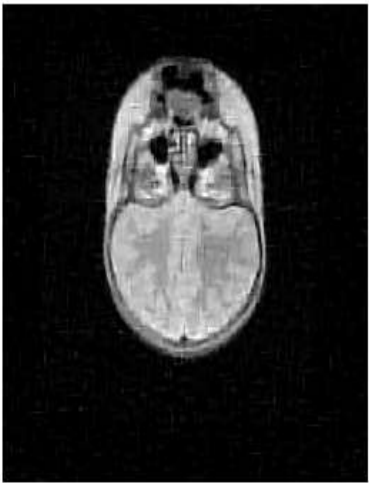
**COMPARISON OF REAL IMAGE, BLURRED (MOTION)  
IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -  
1%)-for different dictionaries**


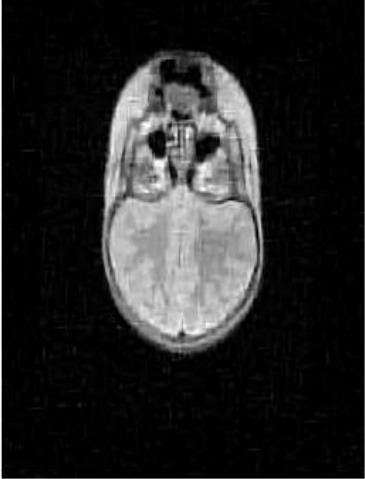


REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.30.COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -1%)-for different dictionaries*


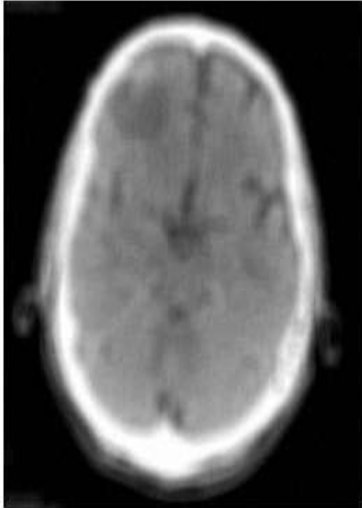


**COMPARISON OF REAL IMAGE, BLURRED (MOTION)  
IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO -  
5%)-for different dictionaries**



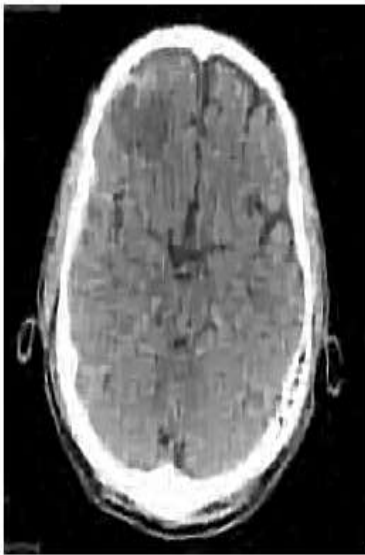

REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.31.COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (MRI 4) (NOISE RATIO -5%)-for different dictionaries*

**COMPARISON OF REAL IMAGE, BLURRED (MOTION)  
IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -  
5%)-for different dictionaries**

REAL IMAGE	BLURRED IMAGE
	
BPG_λ8	BPG_λ1
	

KSVD_r8	KSVD_r6
	
OLM_16	OLM_32
	

*Table.6.32.COMPARISON OF REAL IMAGE, BLURRED (MOTION) IMAGE AND DEBLURRED IMAGE (CT 2) (NOISE RATIO -5%)-for different dictionaries*

### **6.5 INTELLIGENT METHOD:**

In this thesis, an intelligent method has also been studied. This is an extensively study how the performances of these three dictionary algorithms will vary with different choices of an important free parameter i.e. the entries in a weight vector  $\mathbf{w}$  and then finally has proposed an intelligent method of a suitable choice of the weight vector  $\mathbf{w}$  for different deblurring of different medical images to get the best result. The algorithm of this intelligent method implemented for varying the entries of  $\mathbf{w}$  and automatically arriving at the optimum performance is described below in Algorithm 6.1. Once a value of a  $w(i,j)$  element is chosen, that value is chosen identical for all entries in the vector  $\mathbf{w}$ . Next, when a new value of  $\mathbf{w}$  vector has to be chosen, it means that the new value is chosen to be identical for all  $w(i,j)$  elements. And this process is followed in all iterations. In each iteration, the value of  $w(i,j)$  is incremented from 0.1 to 50, with a step size of 0.1. Then for each such chosen value of  $w(i,j)$ , the entire deblurring algorithm is executed and the corresponding PSNR value is calculated. If the PSNR value exceeds 25, then the corresponding PSNR value and its corresponding  $w(i,j)$  values are stored in two different arrays, named as  $P\_array$  and  $w\_array$  respectively. Then from the  $P\_array$  the maximum value of the individual  $p$  is determined and the corresponding value of  $w(i,j)$  is determined from  $w\_array$ . This  $w(i,j)$  value determined is the optimum value for which the algorithm provides the best performance for the image considered for deblurring purpose, with the chosen dictionary learning algorithm.

---

#### ***Algorithm 6.1: Intelligent Method***

---

**INITIALIZATION:** choose  $p$  (PSNR value) as zero, and each  $w(i,j)$  element of  $\mathbf{w}$  as 0.1.

Initialize  $P\_array$  and  $w\_array$  as null vectors.

**WHILE**  $w(i,j) \leq 50$

    Execute the image deblurring algorithm with a chosen dictionary learning algorithm and calculate  $p$ .

**IF**  $p \geq 25$

        Append the  $p$  value in  $P\_array$ .

        Append the  $w(i,j)$  value in  $w\_array$ .

**ENDIF**

Increase  $w(i,j)$  by 0.1.

Reset  $p$ .

## ENDWHILE

Determine the maximum value of  $p$  from  $P\_array$ .

Determine the optimum value of  $w(i,j)$  from the corresponding position of  $w\_array$ .

## Return

By proposing, the Algorithm 6.1 best PSNR value derived for specific  $w$ . For different dictionary, the best PSNR value chart is in Table 6.31 and 6.32. for both MRI and CT scan images. In Table 6.31, for mri\_1 image the best PSNR value is 31.1518 for specific value of weight 0.4, which is same as the data provided in Table.6.1. So that the proposed Algorithm 6.1 is successively applicable for every MRI and CT images.

CHART FOR BEST PSNR VALUE FOR SPECIFIC WEIGHT(MRI)							
Dictionary method		BPG_λ8	BPG_λ1	KSVD_r_8	KSVD_r_6	OLM_16	OLM_32
mri_1	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	31.1518	31.1492	30.985	31.0872	31.1614	31.0709
mri_2	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	31.1487	31.228	31.0166	31.098	31.2274	31.1589
mri_3	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	31.3638	31.4929	31.1896	31.4362	31.3646	31.3737
mri_4	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0162	32.0689	31.7414	31.8845	31.9845	31.4584
mri_5	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0044	32.1122	31.7698	31.4245	31.3648	31.4589
mri_6	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0868	32.2868	31.8746	32.8965	32.2364	32.8945
mri_7	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0517	32.0318	31.8418	32.0874	31.9985	31.9992
mri_8	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0515	32.0455	31.8425	31.9847	31.9845	31.123
mri_9	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.2476	32.2876	32.0924	32.8475	31.123	31.8945
mri_10	WEIGHT	0.4	0.4	0.4	0.4	0.4	0.4
	PSNR VALUE	32.0682	32.0982	31.8056	32.3147	30.8923	32.412



CHART FOR BEST PSNR VALUE FOR SPECIFIC WEIGHT(CT)							
Dictionary method		BPG_λ8	BPG_λ1	KSVD_r_8	KSVD_r_6	OLM_16	OLM_32
ct_1	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	28.3004	28.4449	28.0581	28.2764	28.3832	28.3129
ct_2	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	27.6553	27.8591	27.3685	27.6827	27.7748	27.7007
ct_3	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	26.6644	26.8225	26.7487	26.702	26.6564	26.5834
ct_4	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	26.0081	26.1734	25.616	25.8947	25.8966	27.8956
ct_5	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	27.0753	27.2329	26.738	27.6584	26.4589	26.8945
ct_6	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	27.7056	27.7783	27.3441	28.0145	28.4892	26.5894
ct_7	WEIGHT	0.2	0.2	0.4	0.2	0.2	0.2
	PSNR VALUE	25.6688	25.7319	25.2963	27.6695	27.845	25.9457
ct_8	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	26.9717	26.9996	26.5694	27.9845	26.5894	27.175
ct_9	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	27.8507	27.9559	27.5036	27.8541	26.4569	28.012
ct_10	WEIGHT	0.2	0.2	0.2	0.2	0.2	0.2
	PSNR VALUE	28.2023	28.3378	26.4589	27.2364	25.8956	26.1202

*Table.6.33.CHART FOR BEST PSNR VALUE FOR SPECIFIC WEIGHT (MRI) & (CT)*

## 6.6 CONCLUSION:

The experimental result and discussion point are base medical image recovery. This work is very effective for of producing promising result. The entire medical images proposed a high accuracy in deblurring technique.

# CHAPTER-7

## *CONCLUSION*

- CONCLUSION
- FUTURE SCOPE

# CHAPTER-7

## ***7.1 CONCLUSION:***

The study of image deblurring were done in this thesis for medical image. In this work patched based dictionary learning method is used for image recovery. Deblurring image is a practical scenario that has drawn a lot of attention in image processing. The applied proposed deblurring technique is very effective in recovery performances.

Finally, it should be noted that the recovery performances of different dictionary learning algorithm proposed in his thesis depend on choice of free parameters. The study of effect of all free parameters covered in this thesis. Also an intelligent way is investigated for finding best deblurred image depending weight parameter.

## ***7.2 FUTURE SCOPE:***

Further popularly dictionary learning has been applied for image recovery. This study is helpful for image deblurring. Furthermore research work that can be carried out in this study are given as follows,

- Another operation of image recovery such as denoising, super resolution, compressive imaging recovery can be applied for purpose of medical image recovery.
- Also a better way can be proposed to get automatically best deblurred image for different choice of free parameter.
- Change of patch size and dictionary size could reveal faster and more effective result.

## REFERENCES

- [1] R. C. Gonzalez and R. E. Woods, “Digital Image Processing,” *3rd Edition, Prentice Hall*, (2008)
- [2] Y. Xu and W. Yin, “A fast patch-dictionary method for whole image recovery,” Online code Available: [https://www.caam.rice.edu/~optimization/whole\\_image\\_recon/](https://www.caam.rice.edu/~optimization/whole_image_recon/)
- [3] Y. Xu and W. Yin, “A fast patch-dictionary method for whole image recovery,” *American Institute of Mathematical Sciences*, vol. 10, no. 2, pp. 563-583, 2016.
- [4] Y. Xu and W. Yin, “A fast patch-dictionary method for whole image recovery,” Available: <https://arxiv.org/abs/1408.3740>, 2014.
- [5] M. Elad and M. Aharon, “Image denoising via sparse and redundant representations over learned dictionaries,” *IEEE Transactions on Image Processing*, vol. 15, no. 12, pp. 3736–3745, 2006.
- [6] M. Ben-Ezra and S. K. Nayar, “Motion-based motion deblurring,” *IEEE Trans. PAMI*, vol. 26, no. 6, pp. 689– 698, Jun. 2004
- [7] S. Tiwari, V. P. Shukla, A. K. Singh, S. R. Biradar, “Review of motion blur estimation Techniques,” *Journal of Image and Graphics* Vol. 1, No. 4, December 2013.
- [8] Jian-Feng Cai, Hui Ji, Chaoqiang Liu and Zuowei Shen, “Blind motion deblurring from a single image using sparse approximation,” 978-1-4244-3991-1/09/\$25.00 ©2009 IEEE
- [9] MIDAS insight journal. Dataset for brain MRI. Data Set. Accessed: Nov., 2018. [Online]. Available: <https://www.insight-journal.org/midas/collection/view/171>
- [10] CITUSDATA Azure Database for CT scan of brain. Data Set Accessed: Nov., 2018. [Online]. Available: <https://www.ctisus.com/responsive/teachingfiles/neuro>.
- [11] Y Zhang, J Yang, and Wotao Yin, YALL1: Your algorithms for l1, MATLAB software, <http://yall1.blogs.rice.edu/>, (2010).
- [12] G. Anbarjafari and H. Demirel, “Image super resolution based on interpolation of wavelet domain high frequency subbands and the spatial domain input image,” *ETRI J*, 32 (2010), pp. 390–394.
- [13] S. Ravishankar and Y. Bresler, “Mr image reconstruction from highly undersampled k-space data by dictionary learning,” *Medical Imaging, IEEE Transactions on*, 30 (2011), pp. 1028–1041

- [14] M. Aharon, M. Elad, and A. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *Signal Processing, IEEE Transactions on*, 54 (2006), pp. 4311–4322.
- [15] J. Mairal, F. Bach, J. Ponce, and G. Sapiro, "Online dictionary learning for sparse coding," *In Proceedings of the 26<sup>th</sup> Annual International Conference on Machine Learning, ACM*, 2009, pp. 689–696.
- [16] Y. Xu and W. Yin, "A block coordinate descent method for regularized multi-convex optimization with applications to nonnegative tensor factorization and completion," *To appear in SIAM Journal on Imaging Science*, (2013).
- [17] P. De, A. Chatterjee, and A. Rakshit, "Recognition of Human Behavior for Assisted Living Using Dictionary Learning Approach," *IEEE Sensor journal*, VOL. 18, NO. 6, March 15, 2018, pp. 2434–2440.
- [18] Tosić and P. Frossard, "Dictionary learning, Signal Processing Magazine," *IEEE*, 28 (2011), pp. 27–38.
- [19] P. Tseng, "Convergence of a block coordinate descent method for non differentiable minimization," *Journal of Optimization Theory and Applications*, 109 (2001), pp. 475–494.
- [20] M. Elad, "Sparse and redundant representations: From Theory to Applications in Signal and Image Processing," *Springer*, 2009.
- [21] M. Elad, "Sparse Representations Are Most Likely to Be the Sparsest Possible," *EURASIP Journal on Applied Signal Processing*, volume 2006, pp. 1–12.
- [22] S. Shaobing Chen<sup>†</sup>, David L. Donoho<sup>‡</sup>, Michael A. Saunders<sup>§</sup>, "Atomic Decomposition by Basis Pursuit<sup>\*</sup>," Vol. 43, No. 1, pp. 129–159, 2001.
- [23] Stephane G. Mallat, Z. Zhang, "Matching Pursuits with time frequency dictionaries," *IEEE Transactions*, vol. 41, Dec 1993, pp. 3397–3415.
- [24] M. Ben-Ezra and S. K. Nayar, "Motion-based motion deblurring," *IEEE Trans. PAMI*, vol. 26, no. 6, pp. 689–698, Jun. 2004.
- [25] P. Ganesan, G. Sapiro, "Deblurring techniques of Natural Images," *Third international conference and management, IEEE*, 2017.
- [26] M. M. Bronstein, A. M. Bronstein, M. Zibulevsky, and Y. Y. Zeevi, "Blind deconvolution of images using optimal sparse representations," *IEEE Trans. Image Process.*, vol. 14, no. 6, pp. 726–736, Jun. 2005.
- [27] Hui-yu Huang, Wei-chang Tsai, "Motion Deblurring from a Single Photograph Based on Kernel Estimation," *ICICS 2013, IEEE*.

- [28] M. Aharon, M. Elad, "Sparse and redundant modeling of image content using an image-signature dictionary," *SIAM Imaging Sciences*, vol. 1, pp. 228–247, 2008.
- [29] M. Protter, M. Elad, "Image sequence denoising via sparse and redundant representations." *IEEE Transactions Image Processing*, vol. 18, pp. 27–36, 2009.