Sensing Matrix Design Techniques

BTP-2 Report

Ameya Anjarlekar
170070013

Supervised By

Prof. Ajit Rajwade and Prof. Manoj Gopalkrishnan

Department of Electrical Engineering
Indian Institute of Technology, Bombay
Powai, Mumbai - 400 076.

May 13, 2021

Contents

1 Introduction

1.1 Compressed Sensing

Compressed sensing is a signal processing technique for efficiently acquiring and reconstructing a signal, by finding solutions to underdetermined linear systems. The signals are generally undersampled below Nyquist rate and then a lossy reconstruction is performed. For reconstruction, the sparsity of signals in certain bases is exploited. Generally for images, these bases could be DCT (Discrete Cosine Transform), Wavelet, Fourier or even dictionaries learned using training data. For reconstruction, algorithms such as OMP or LASSO are generally used.
Let x be the signal and Φ be the sensing matrix. Then $y = \Phi \cdot x$ represent the measurements which we obtain. Since x is a sparse signal in some basis, let $x = D\theta$ where D is the basis and θ is the sparse vector. Then the compressed sensing problems tries to find θ using y and $A = \Phi \cdot D$ [1].

1.2 Mutual Coherence and RIP

For efficient compressive recovery, the choice of dictionary and sensing matrix should satisfy certain conditions. CS theory states that Φ and D should be "incoherent" with each other. The coherence between Φ and D is defined as

$$\mu(\Phi, D) = \sqrt{n} \cdot max_{1 \leq j \leq m, 1 \leq i \leq n} | < \Phi^j, D_i > | \tag{1}$$

where Φ^j is the j^{th} row of Φ and D_i is the i^{th} column of D. We need this coherence to be as minimum as possible. However, the bounds provided by mutual coherence are very loose. Moreover [2] also highlights some issues with using maximum mutual coherence and shows better results with average mutual coherence. To get tighter bounds, RIP (Restricted Isometric Property) [3] can be used. For integer S = 1,2,. . . ,n, the restricted isometry constant (RIC) δ_s of a matrix A of size m by n is the smallest number such that for any S-sparse vector θ, the following holds

$$(1 - \delta_s)||\theta||^2 \leq ||A \cdot \theta||^2 \leq (1 + \delta_s)||\theta||^2 \tag{2}$$

This provides tighter bounds, however calculations of RIC are computationally expensive.

1.3 Reconstruction Algorithms

The compressed sensing reconstruction problem aims to solve the following optimization problem

$$min||\theta||_0 \quad st \quad ||Y - A\theta||_2^2 \leq \epsilon \tag{3}$$

This is an NP-hard problem. Thus, approximation algorithms such as Orthogonal Matching Pursuit (OMP) or Co-SAMP are proposed.
Another alternative is to minimize the L_1 norm [4] instead of the L_0 norm. Commonly used L_1-optimization algorithms include LASSO or Basis Pursuit.
Lasso optimization problem is given as

$$min||Y - A\theta||_2^2 \quad st \quad ||\theta_0|| \leq \tau \tag{4}$$

Basis Pursuit is defined as

$$min||\theta||_1 \quad st \quad ||Y - A\theta||_2^2 \leq \epsilon \tag{5}$$

2 Sensing matrix design using the $l_1 - l_\infty$ measure

Various learning algorithms are proposed in literature for efficient sensing matrix design. [5] uses training data to estimate the position of atoms for a given image reconstruction problem. The weighted Mutual Coherence algorithm designed in [6] proposes a way to effectively reconstruct video frames by using a weighted mutual coherence approach through designing weights from preceding video frames. We take inspiration from this work to propose our weighted coherence approaches where the weights are taken from training data.
[2] [7] [8] proposes an approach which uses the L_∞ norm to develop verifiable sufficient conditions and computable performance bounds of L_1-minimization based sparse recovery algorithms. We extend this work to develop faster bounds using L_0-minimization based sparse recovery algorithms. The paper proposes that: Suppose x is k-sparse and the noise w satisfies $||w||_2 \leq \epsilon$ and $\hat{x}$ is the optimum obtained from the basis pursuit denoising. Then,

$$||\hat{x} - x||_\infty \leq \frac{2\epsilon}{\omega_2(A, 2k)} \tag{6}$$

and

$$\omega_2(Q, s) = min_{z:\frac{||z||_1}{||z||_\infty} \leq s} \frac{||Q \cdot z||_2}{||z||_\infty} \tag{7}$$

For calculating $\omega_2(Q, s)$ the following minimization problem is proposed in [2]

$$\omega_2(Q, s) = min_{i,\lambda}||Q_i - Q(:, -i) \cdot \lambda||_2 \quad s.t.||\lambda||_1 \leq s - 1 \tag{8}$$

where s is a chosen hyperparameter and $Q(:, -i)$ denotes all the columns of Q except the i^{th} column.

3 Proposed Coherence based approaches

3.1 Without training data

We propose coherence minimization based approaches for matrix design. The main intuition behind the approach is that any column vector of the sensing matrix should not be able to be represented as a linear combination of other column vectors. This ensures that during sparse recovery the picking of the non-zero atoms becomes easier because the sensing matrix column corresponding to it is hard to replace using other column vectors.
Hence, based on this idea, we propose coherence, bi-coherence and tri-coherence approaches.

1) **Coherence**

$$A = argmax_A \sum_i \sum_{j>i} argmin_{\alpha(i,j)} ||A_i - \alpha(i,j)A_j||_2^2 \tag{9}$$

where A_i denotes the i^{th} column of A
The minimization step can be solved by assigning $\alpha(i,j) = (A_j' \cdot A_j)^{-1} \cdot A_j' \cdot A_i$
Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} ||A_i - (A_j'A_j)^{-1}A_j'A_iA_j||_2^2 \tag{10}$$

2) **bi-Coherence**

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} argmin_{\alpha(i,j,k)} ||A_i - \alpha(i,j,k)A_{j,k}||_2^2 \tag{11}$$

where A_i denotes the i^{th} column of A and the matrix $A_{j,k}$ matrix formed by concatenating the j^{th} and k^{th} columns of A.
The minimization step can be solved by assigning $\alpha(i,j,k) = (A_{j,k}' \cdot A_{j,k})^{-1} \cdot A_{j,k}' \cdot A_i$
Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} ||A_i - (A_{j,k}'A_{j,k})^{-1}A_{j,k}'A_iA_{j,k}||_2^2 \tag{12}$$

2) **tri-Coherence**

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} \sum_{l>k} argmin_{\alpha(i,j,k,l)} ||A_i - \alpha(i,j,k,l)A_{j,k,l}||_2^2 \tag{13}$$

where A_i denotes the i^{th} column of A and the matrix $A_{j,k,l}$ matrix formed by concatenating the j^{th}, k^{th} and l^{th} columns of A.
The minimization step can be solved by assigning $\alpha(i,j,k,l) = (A_{j,k,l}' \cdot A_{j,k,l})^{-1} \cdot A_{j,k,l}' \cdot A_i$

Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} \sum_{l>k} ||A_i - (A'_{j,k,l} A_{j,k,l})^{-1} A'_{j,k,l} A_i A_{j,k,l}||_2^2 \tag{14}$$

3.2 Using training data

We also propose weighted coherence based approaches for cases in which training data is available. The main idea behind it is that certain frequencies are more likely to be used over other frequencies to represent natural images. Hence, we assign weights to emphasize certain groups of column vectors over other column vectors.
To calculate the weights (W) we represent the training data by a sparse basis (DCT in our case) and take the mean.

$$W = \sum_{i=1}^{N} \frac{1}{N} D \cdot X_i(:) \tag{15}$$

where $X_i(:)$ is the i^{th} training image in vectorized form and D is the 2d-DCT matrix. Thus, W is a vector of weights where each element represents weight of every frequency. Using the weights we propose weighted coherence, weighted bi-coherence/double coherence and weighted tri-coherence/triple coherence

1) **Weighted Coherence**

$$A = argmax_A \sum_i \sum_{j>i} argmin_{\alpha(i,j)} \sqrt{W(i) \cdot W(j)} ||A_i - \alpha(i,j) A_j||_2^2 \tag{16}$$

where W is the weight matrix obtained from training data and A_i denotes the i^{th} column of A

The minimization step can be solved by assigning $\alpha(i,j) = (A'_j \cdot A_j)^{-1} \cdot A'_j \cdot A_i$
Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} \sqrt{W(i) \cdot W(j)} ||A_i - (A'_j A_j)^{-1} A'_j A_i A_j||_2^2 \tag{17}$$

2) **Weighted bi-Coherence**

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} argmin_{\alpha(i,j,k)} (W(i) \cdot W(j) \cdot W(k))^{\frac{1}{3}} ||A_i - \alpha(i,j,k) A_{j,k}||_2^2 \tag{18}$$

where W is the weight matrix obtained from training data, A_i denotes the i^{th} column of A, and the matrix $A_{j,k}$ matrix formed by concatenating the j^{th} and k^{th} columns of A.

The minimization step can be solved by assigning $\alpha(i,j,k) = (A'_{j,k} \cdot A_{j,k})^{-1} \cdot A'_{j,k} \cdot A_i$

Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} (W(i) \cdot W(j) \cdot W(k))^{\frac{1}{3}} ||A_i - (A'_{j,k} A_{j,k})^{-1} A'_{j,k} A_i A_{j,k}||_2^2 \tag{19}$$

2) **Weighted tri-Coherence**

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} \sum_{l>k} argmin_{\alpha(i,j,k,l)} (W(i) \cdot W(j) \cdot W(k) * W(l))^{\frac{1}{4}} ||A_i - \alpha(i,j,k,l) A_{j,k,l}||_2^2 \tag{20}$$

where W is the weight matrix obtained from training data, A_i denotes the i^{th} column of A, and the matrix $A_{j,k,l}$ matrix formed by concatenating the j^{th}, k^{th} and l^{th} columns of A.

The minimization step can be solved by assigning $\alpha(i,j,k,l) = (A'_{j,k,l} \cdot A_{j,k,l})^{-1} \cdot A'_{j,k,l} \cdot A_i$
Thus, the following optimization problem results

$$A = argmax_A \sum_i \sum_{j>i} \sum_{k>j} \sum_{l>k} (W(i) \cdot W(j) \cdot W(k) * W(l))^{\frac{1}{4}} ||A_i - (A'_{j,k,l} A_{j,k,l})^{-1} A'_{j,k,l} A_i A_{j,k,l}||_2^2 \tag{21}$$

4 Weighted basis pursuit recovery

Finally, we propose a modified basis pursuit recovery algorithm by using the weights obtained through training data. We emphasize certain atoms during recovery using this modified optimization problem

$$min||W^{-1}\theta||_1 \quad st \quad ||y - A\theta||_2 \leq \sigma \tag{22}$$

where the hyperparameter σ is chosen through cross-validation

5 Observations and results

5.1 Evaluations

We perform reconstructions on the compressive measurements from cropped images of the Berkley dataset (image size = 160x240). The Patch size is taken to be 10x10 and iid gaussian noise is added to the measurements with std deviation = 2. DCT basis is used for sparse representation. A block based sparse reconstruction is preferred [9] instead of the Rice single pixel approach [10] owing to the large image size. The experiments are performed by considering the sensing matrices values to be restricted between (0,1) (the random matrix is chosen to be U(0,1)) in the first case and using binary sensing matrices in the second case. More emphasis is placed on binary sensing matrix design because of its use in Group testing [11]. Reconstruction is done on 100 images while the weights are generated using 200 training images. Basis pursuit algorithm is used for reconstruction using a SPGL solver [12] [13].
For the [0,1] matrices, we use a projected gradient ascent approach for optimizing over the matrices.

For the binary matrices, the algorithm is as follows

Algorithm 1 Binary matrix optimization

take ϕ as random binary matrix
for $i = 1 : m$ **do**
  for $j = 1 : n$ **do**
    $l_1 = L(\phi)$ $\phi(i,j) = 1 - \phi(i,j)$ $l_2 = L(\phi)$
    if $l_2 \leq l_1$ **then**
      $\phi(i,j) = 1 - \phi(i,j)$
    else
  end
end

5.2 U(0,1) matrices

Uniform (0,1) matrices are optimized by maximizing coherence, double coherence, weighted coherence and weighted double coherence. The average PSNR, RRMSE and SSIM values are plotted. We can see that the weighted versions perform better than the ones without training data. Further, matrices optimized using double coherence perform better than the coherence versions.

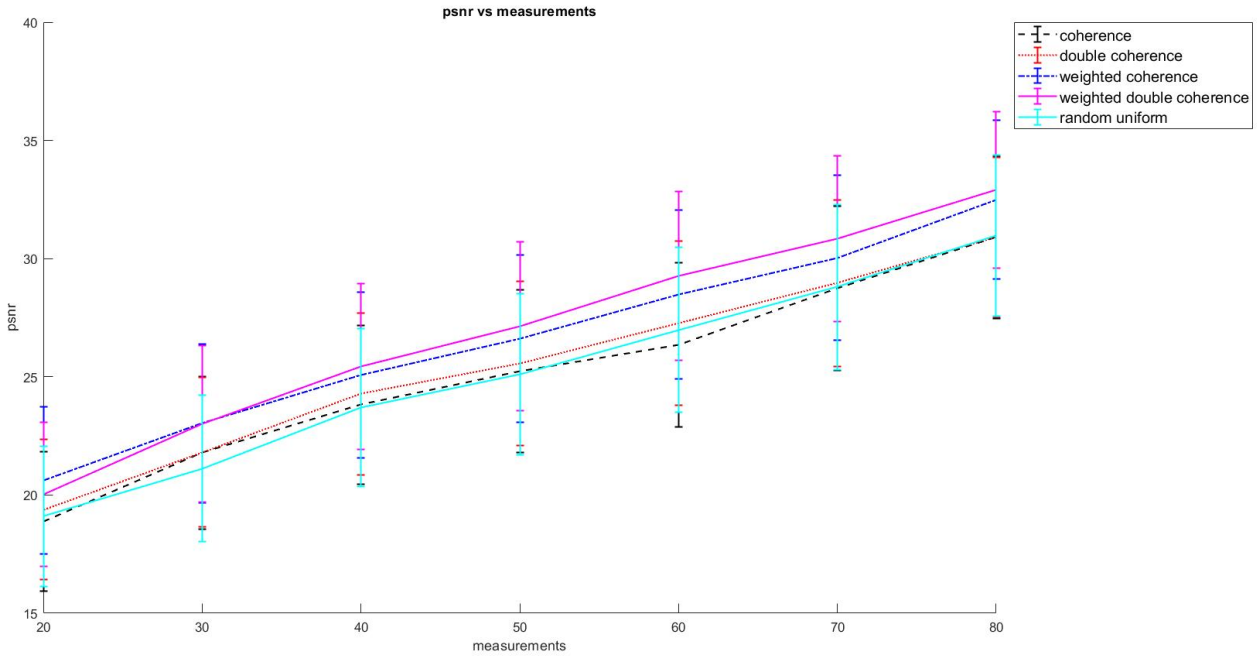

Figure 1: PSNR comparison of different coherence based methods on U(0,1) matrices

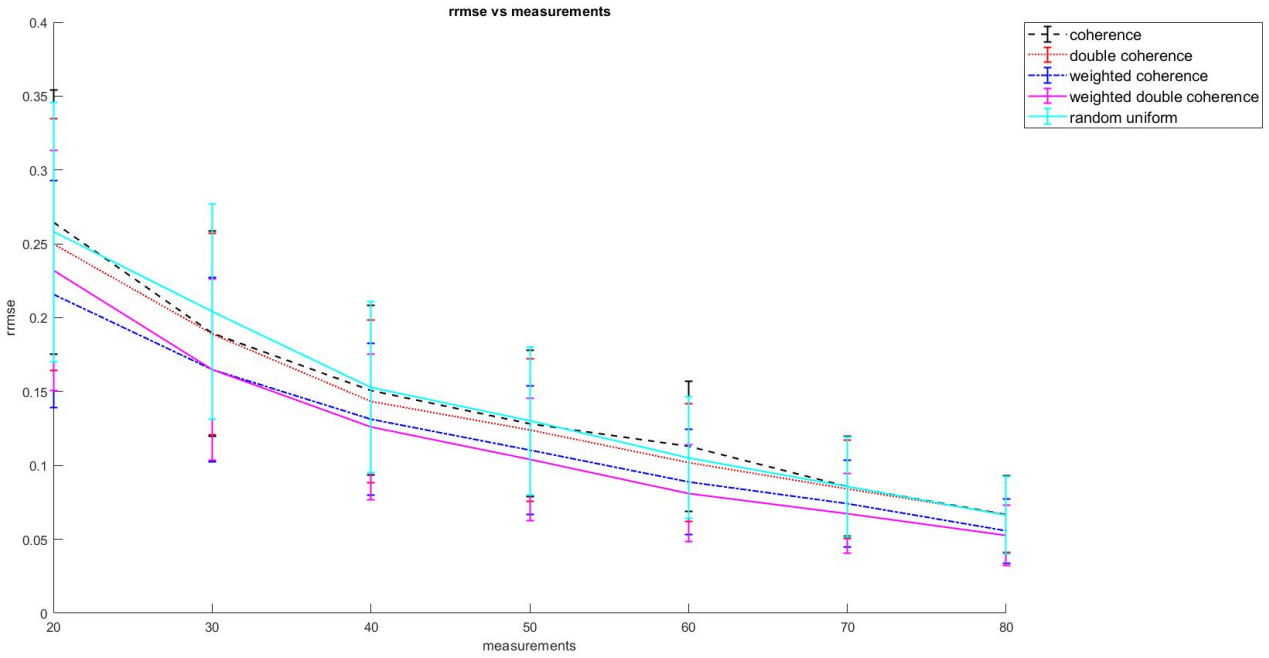

Figure 2: RRMSE comparison of different coherence based methods on U(0,1) matrices

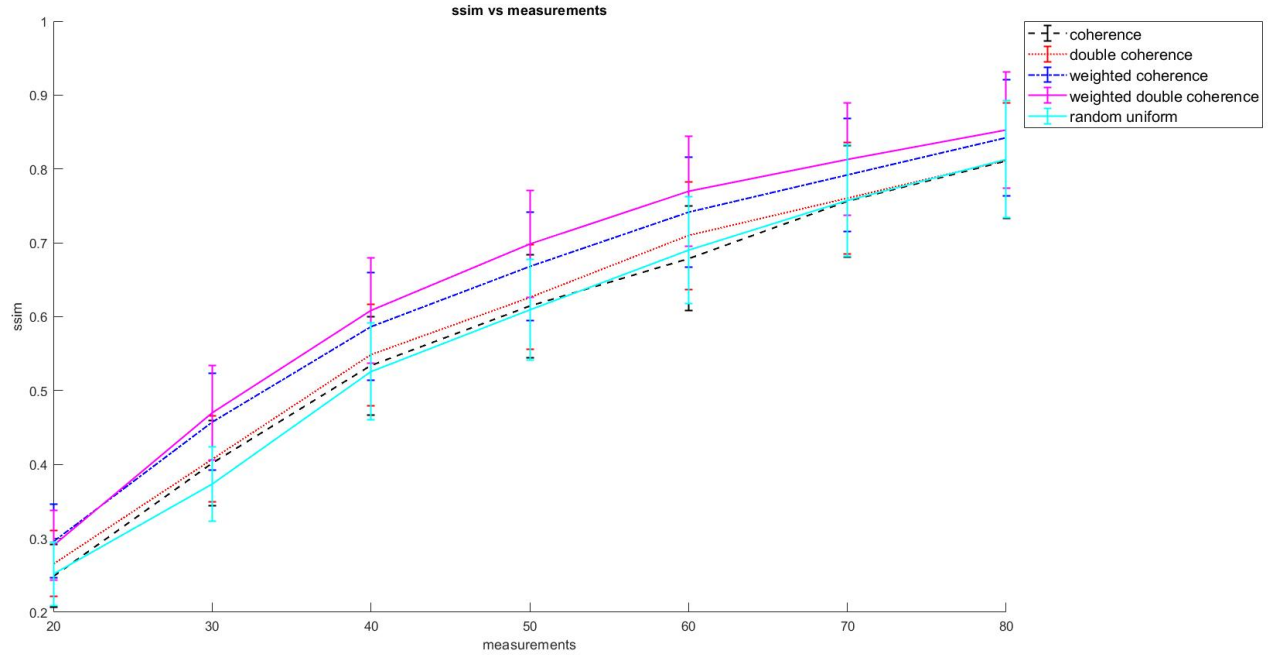

Figure 3: SSIM comparison of different coherence based methods on U(0,1) matrices

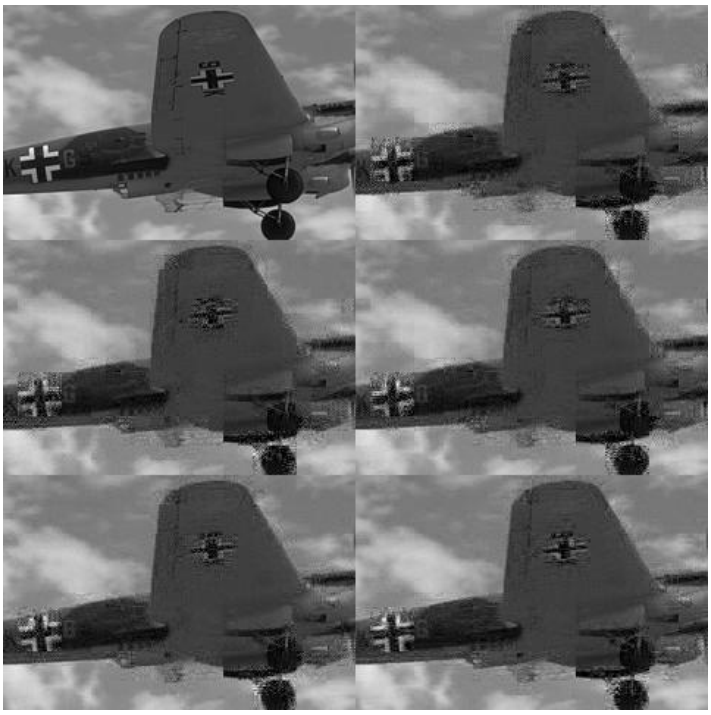

Figure 4: Measurements = 40/100
PSNR Values from top to bottom 27.3088, 27.4607, 27.9987, 28.9305, 29.2511. The images correspond to actual, random uniform, coherence, double coherence, weighted coherence and weighted double coherence.

Figure 5: Measurements = 40/100
PSNR Values from top to bottom 24.4214, 24.6054, 25.0633, 25.8433, 26.2447. The images correspond to actual, random uniform, coherence, double coherence, weighted coherence and weighted double coherence.

Figure 6: Measurements = 50/100
PSNR Values from top to bottom 23.3918, 23.5117, 23.8989, 24.7837, 25.3068. The images correspond to actual, random uniform, coherence, double coherence, weighted coherence and weighted double coherence.

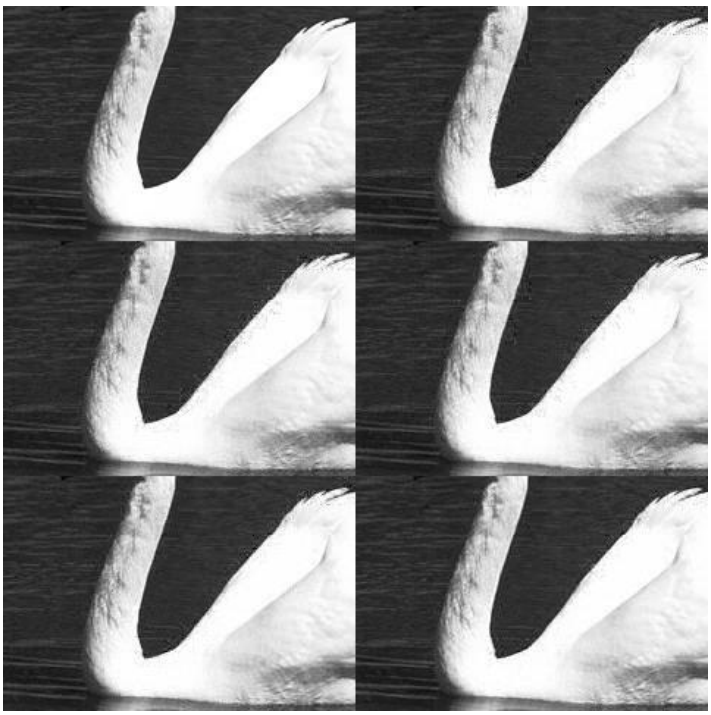

Figure 7: Measurements = 70/100
PSNR Values from top to bottom 32.0536, 32.1411, 32.5653, 33.5493, 34.7726. The images correspond to actual, random uniform, coherence, double coherence, weighted coherence and weighted double coherence.

5.3 Binary Matrices

5.3.1 Coherence methods

Random binary matrices are optimized by maximizing coherence, double coherence, triple coherence (unweighted), weighted coherence and weighted double coherence. The average PSNR, RRMSE and SSIM values are plotted. We can see that the weighted versions perform better than the ones without training data. Further, matrices optimized using double coherence and triple coherence perform better than the coherence version with double and triple coherence optimized matrices having similar performance.

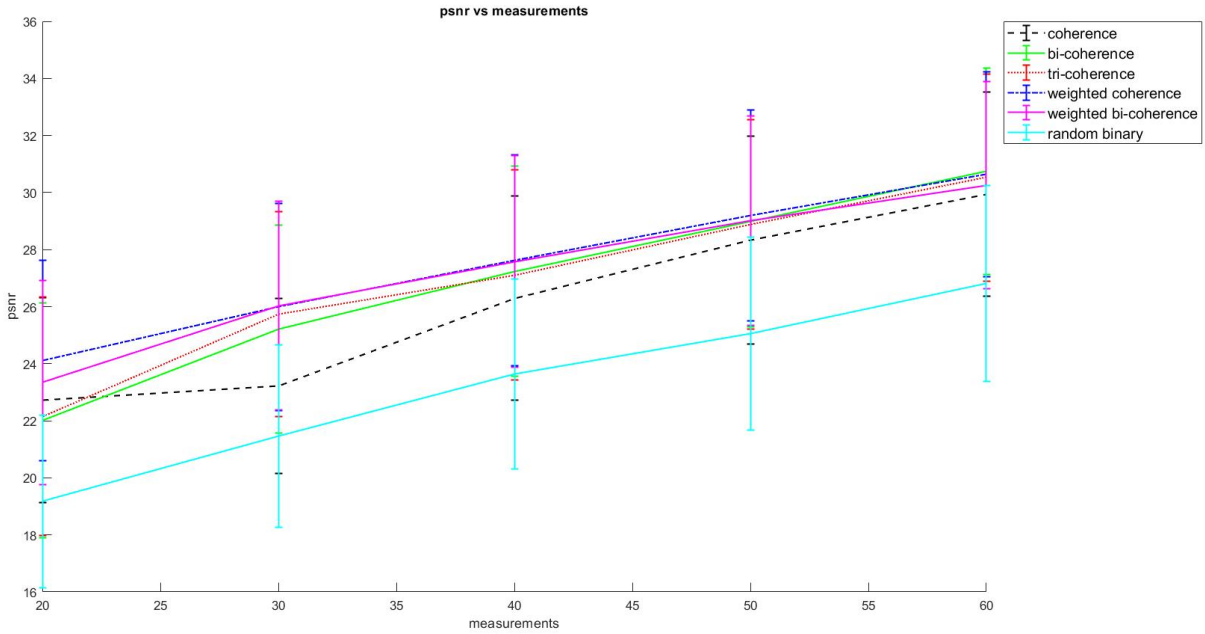

Figure 8: PSNR comparison of different coherence based methods on binary matrices

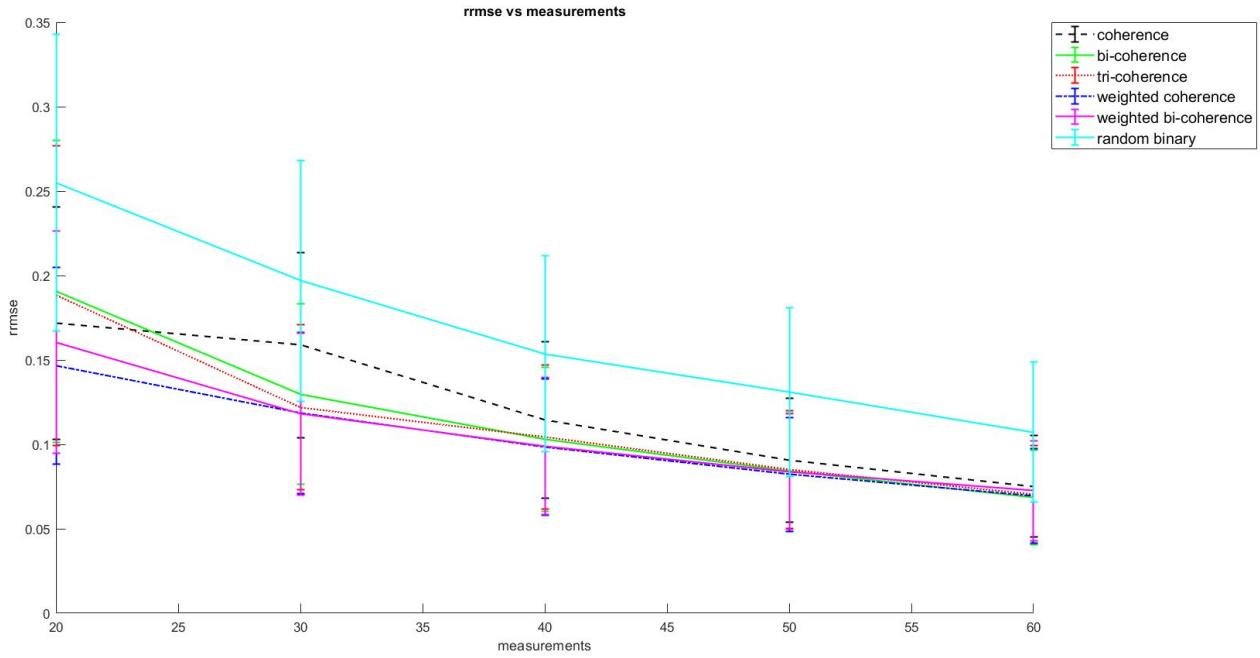

Figure 9: RRMSE comparison of different coherence based methods on binary matrices

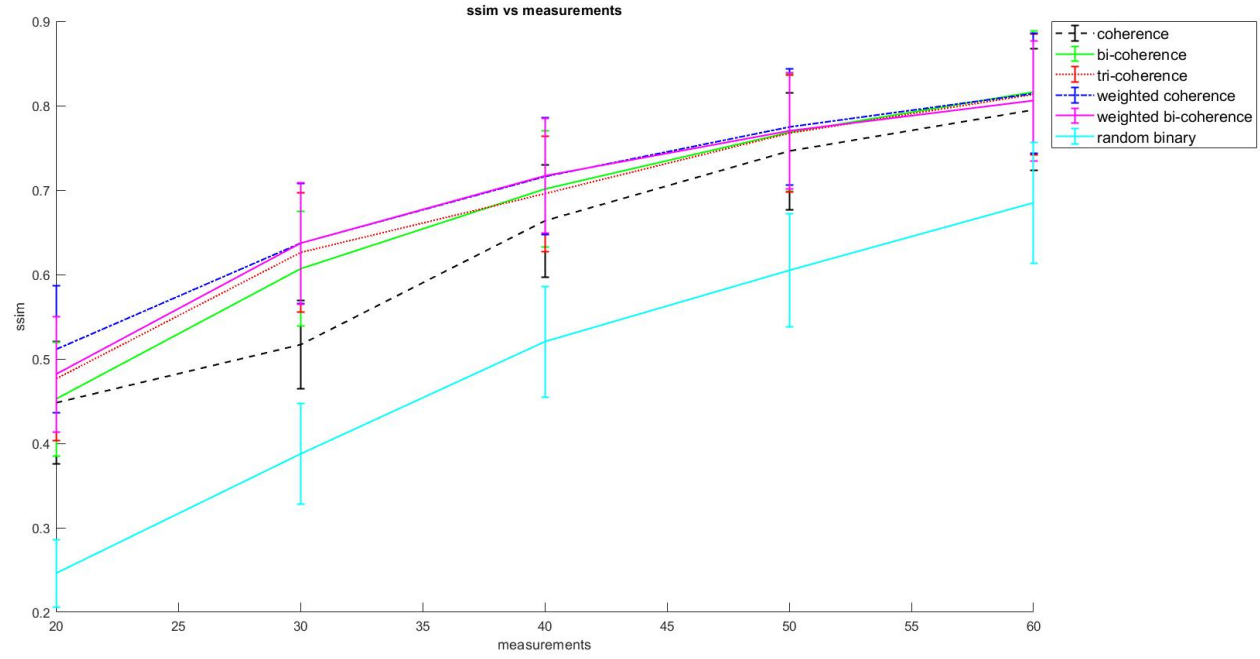

Figure 10: SSIM comparison of different coherence based methods on binary matrices

Figure 11: Actual image

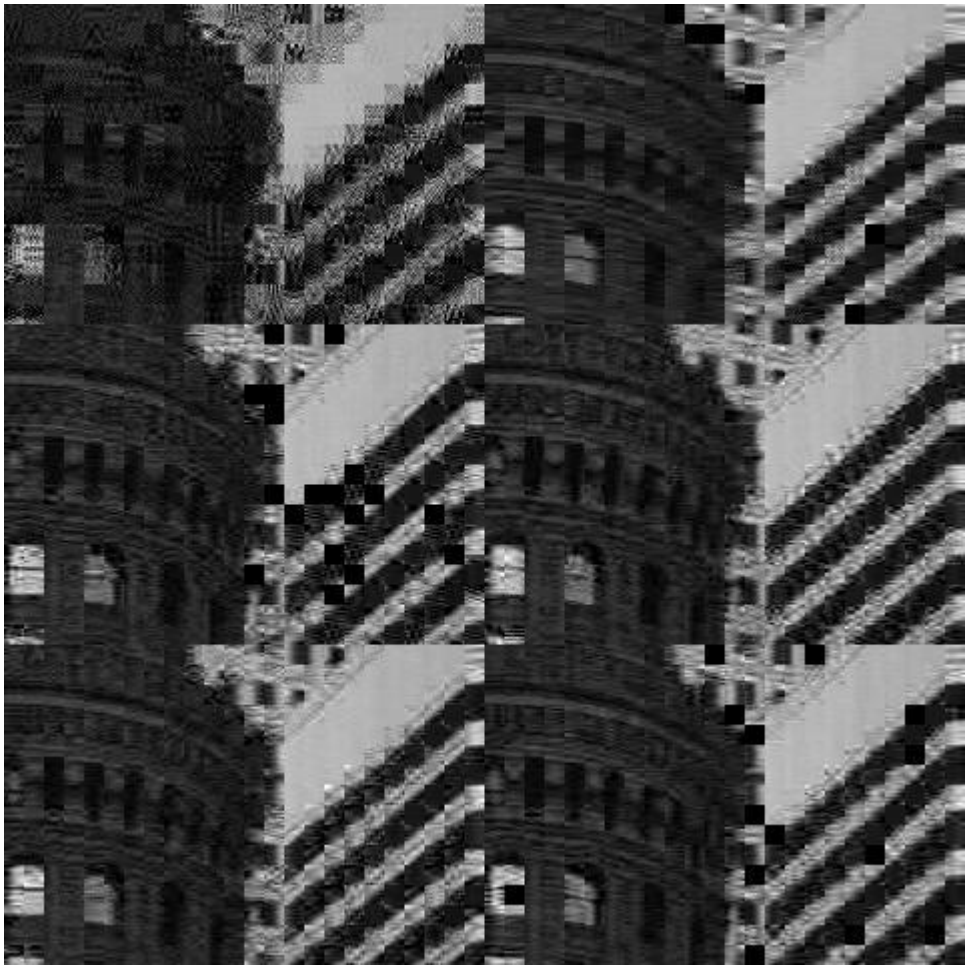

Figure 12: Measurements = 20/100
PSNR Values from top to bottom 16.637, 18.8659, 17.0941, 20.9852, 20.7071, 17.7011. The images correspond to random binary, coherence, double coherence, weighted coherence, weighted double coherence and triple coherence.

Figure 13: Actual image

Figure 14: Measurements = 30/100
PSNR Values from top to bottom 21.7323, 24.7860, 25.569, 26.202, 26.4096, 25.9373. The images correspond to random binary, coherence, double coherence, weighted coherence, weighted double coherence and triple coherence.

Figure 15: Actual image

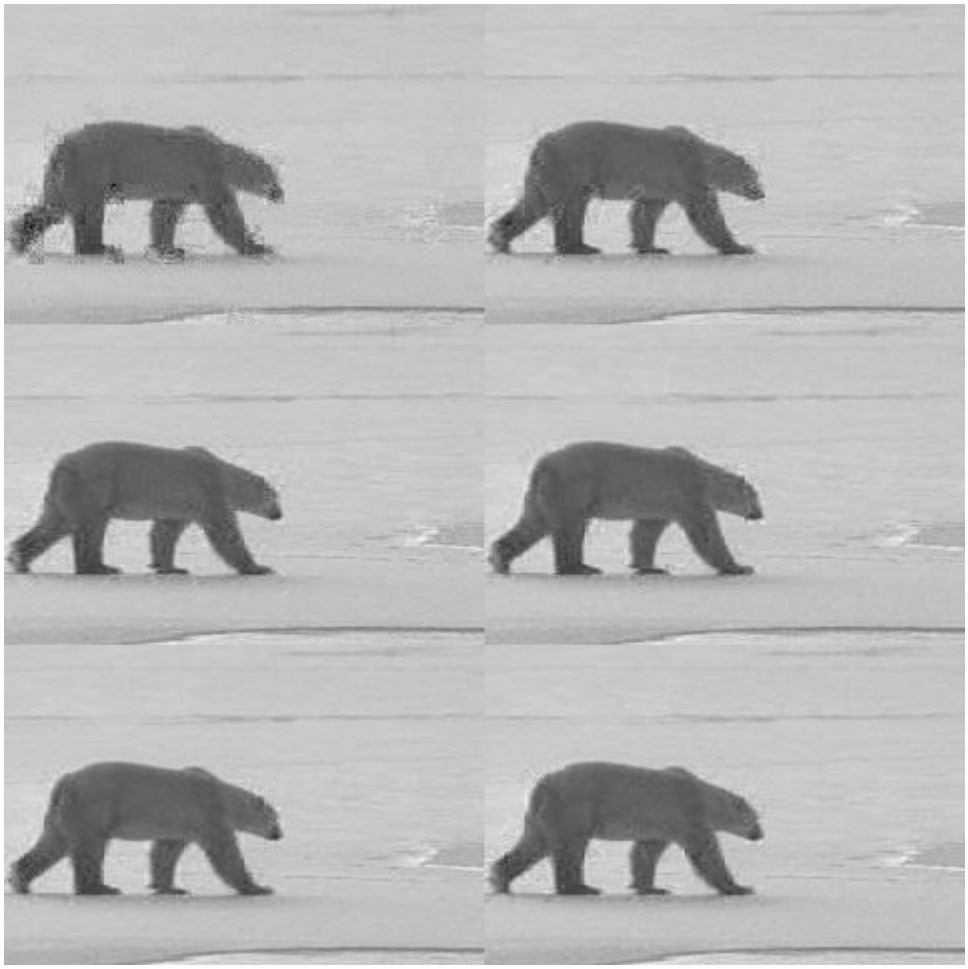

Figure 16: Measurements = 40/100
PSNR Values from top to bottom 30.2151, 33.7755, 35.2245, 35.3347, 35.5187, 34.9959. The images correspond to random binary, coherence, double coherence, weighted coherence, weighted double coherence and triple coherence.

Figure 17: Actual image

Figure 18: Measurements = 50/100
PSNR Values from top to bottom 21.519, 24.6049, 25.3521, 25.2628, 25.1922, 25.3063. The images correspond to random binary, coherence, double coherence, weighted coherence, weighted double coherence and triple coherence.

Figure 19: Actual image

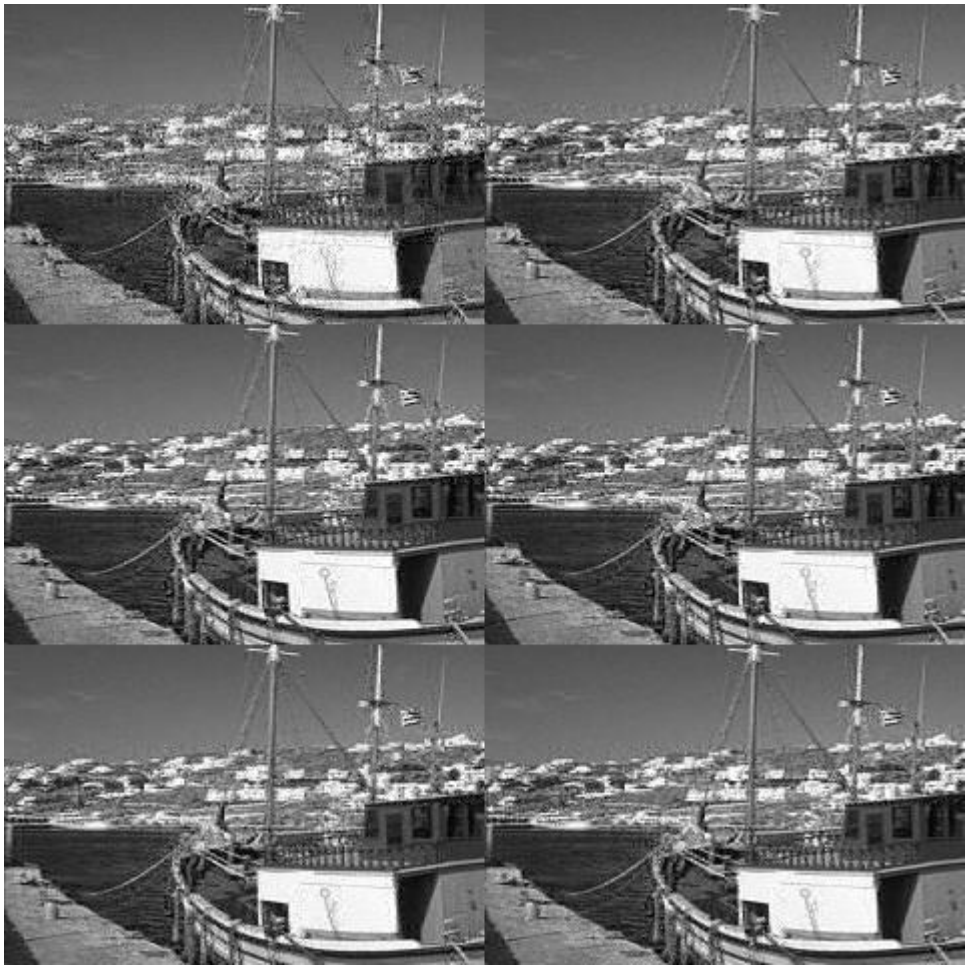

Figure 20: Measurements = 60/100
PSNR Values from top to bottom 22.0842, 24.9541, 25.8342, 25.8175, 25.5950, 25.6432. The images correspond to random binary, coherence, double coherence, weighted coherence, weighted double coherence and triple coherence.

5.3.2 Comparison of non-data driven methods

In this section random binary matrices are optimized using non data-driven techniques such as by maximizing coherence, double coherence, triple coherence and Nehorai's $l_1 - l_\infty$ measure. The hyperparameter s for the $l_1 - l_\infty$ measure is chosen by selecting s with the lowest reconstruction error. The average PSNR, RRMSE and SSIM values are plotted. We can see that the coherence optimized matrices perform better than the ones optimized using the $l_1 - l_\infty$ measure. Further, matrices optimized using triple coherence provide the best results for smaller measurements. As the number of measurements increase, double coherence optimized matrices perform better.

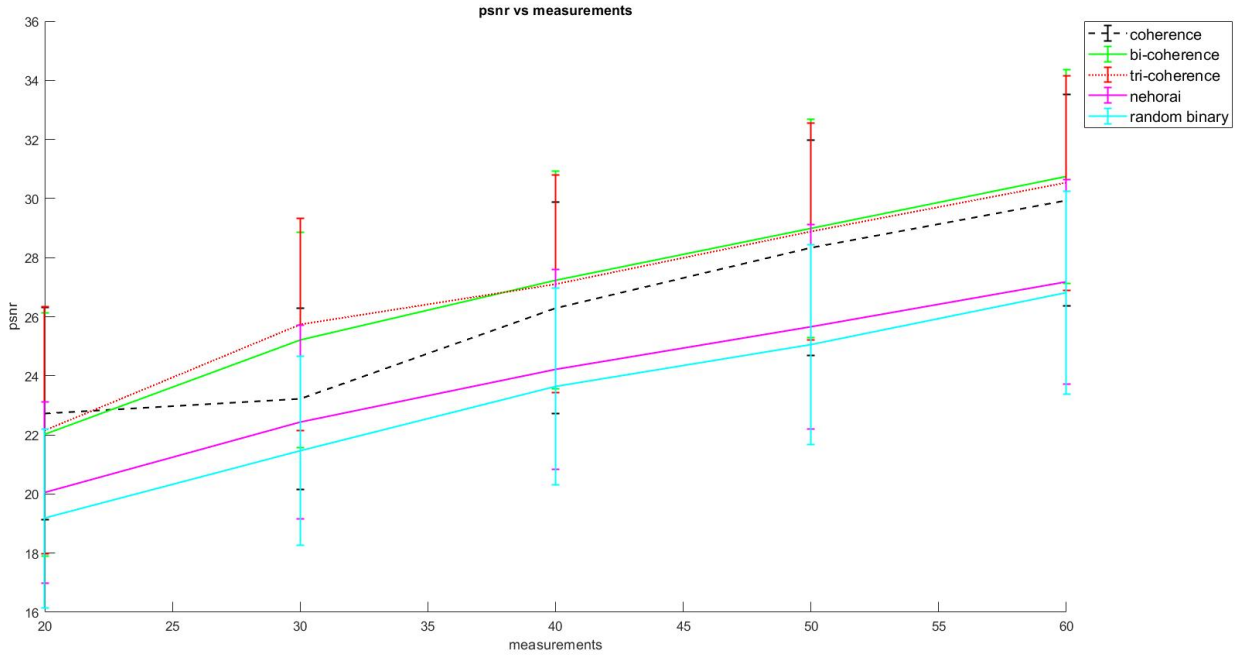

Figure 21: PSNR comparison of different non-data driven matrix optimization methods on binary matrices

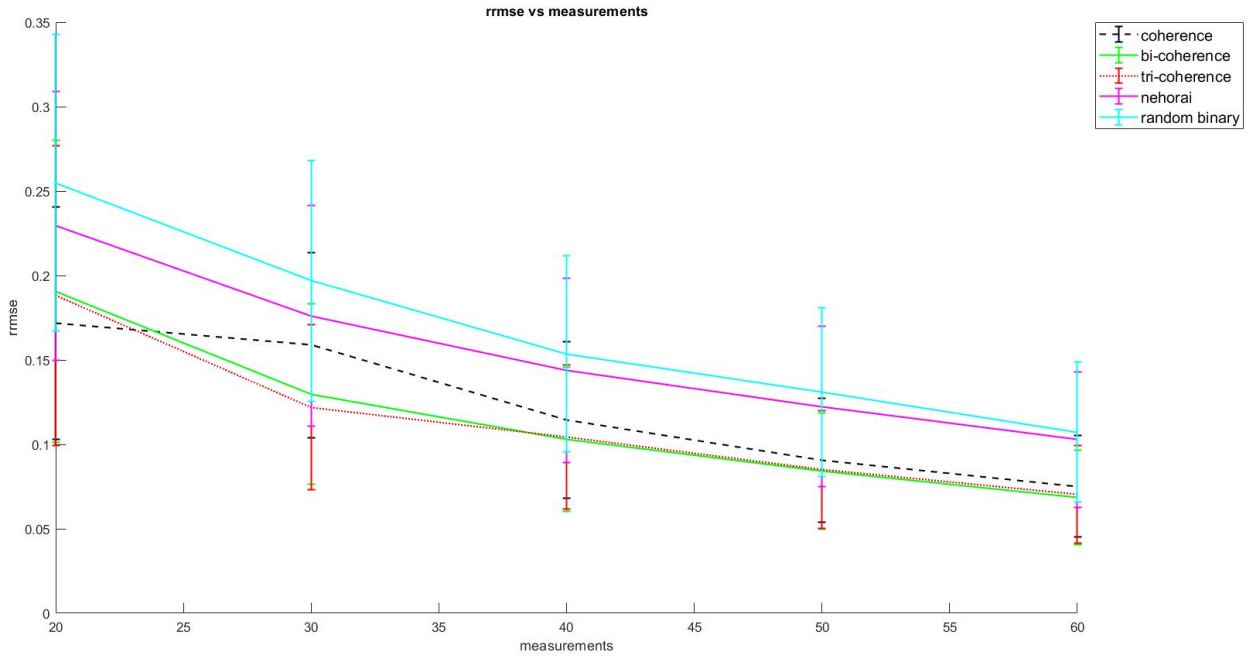

Figure 22: RRMSE comparison of different non-data driven matrix optimization methods on binary matrices

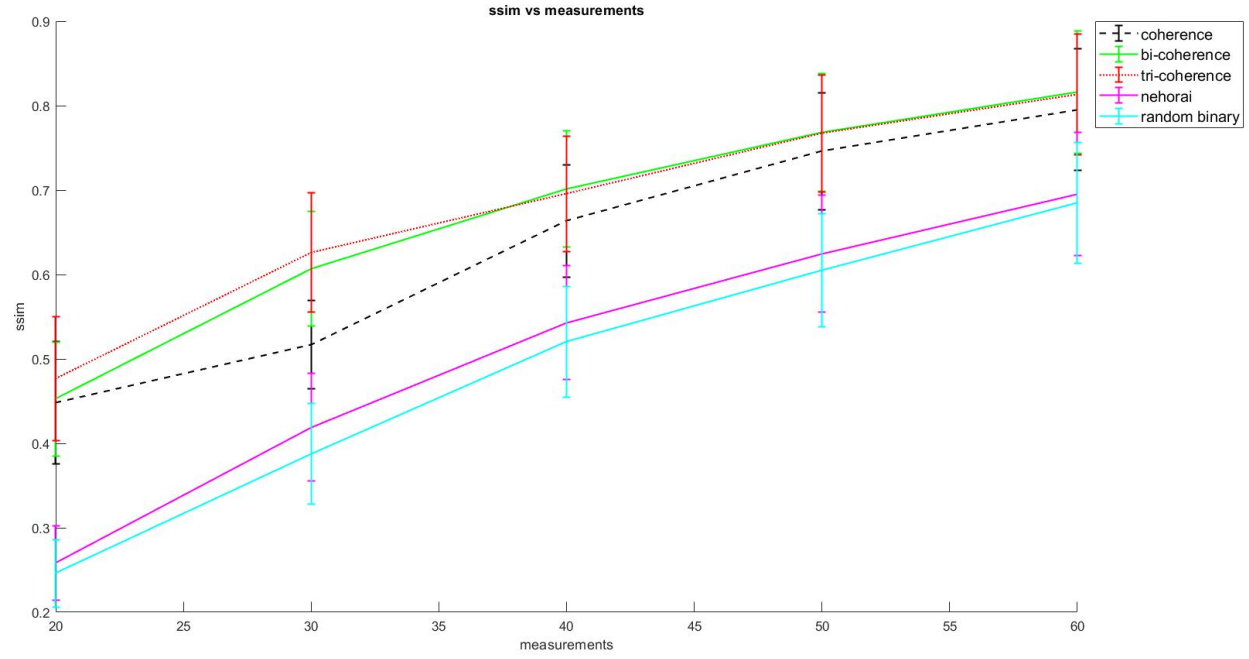

Figure 23: SSIM comparison of different non-data driven matrix optimization methods on binary matrices

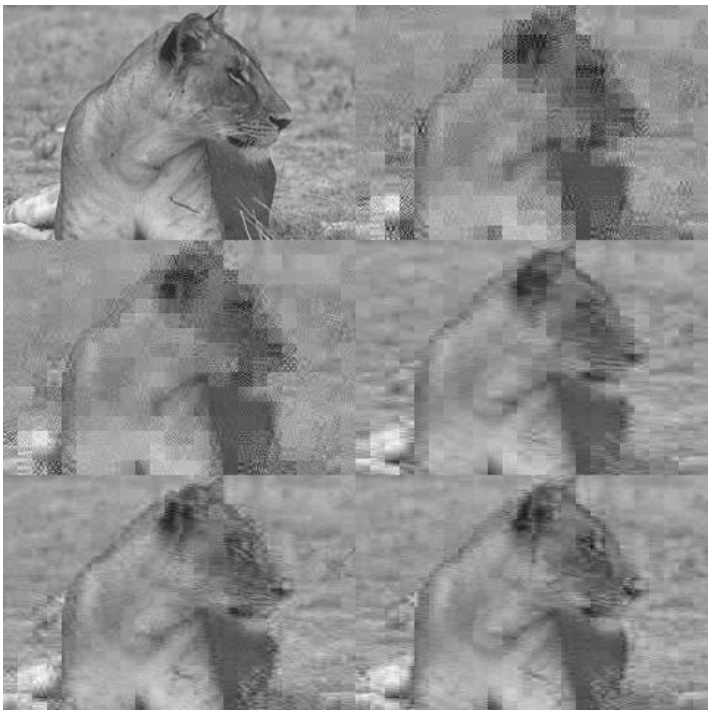

Figure 24: Measurements = 20/100
PSNR Values from top to bottom 22.6455, 23.4731, 26.7143, 26.7683, 27.2991. The images correspond to actual image, random binary, $l_1 - l_\infty$ approach, coherence, double coherence and triple coherence

Figure 25: Measurements = 30/100
PSNR Values from top to bottom 23.4211, 25.4132, 27.678, 29.7609, 29.2127. The images correspond to actual image, random binary, $l_1 - l_\infty$ approach, coherence, double coherence and triple coherence

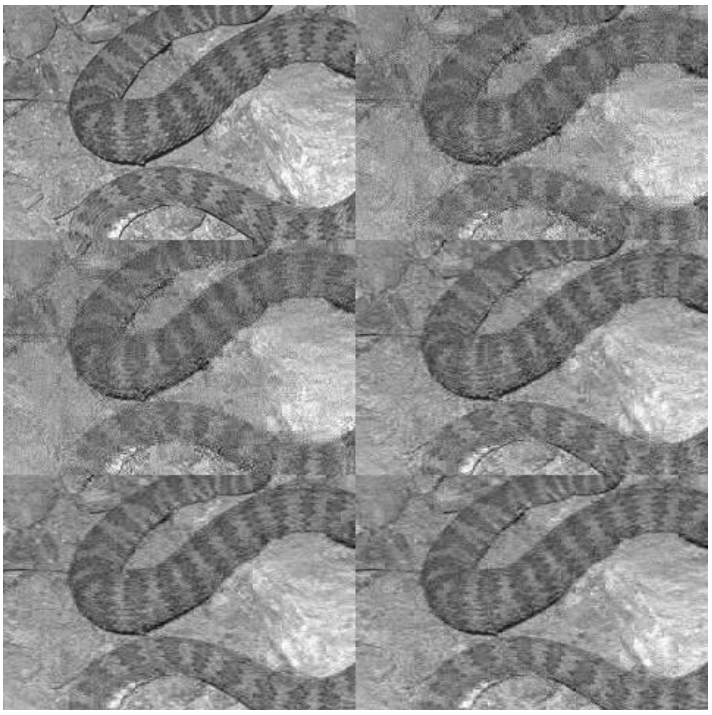

Figure 26: Measurements = 40/100
PSNR Values from top to bottom 22.5332, 23.0011, 25.0203, 25.7331, 25.7704. The images correspond to actual image, random binary, $l_1 - l_\infty$ approach, coherence, double coherence and triple coherence

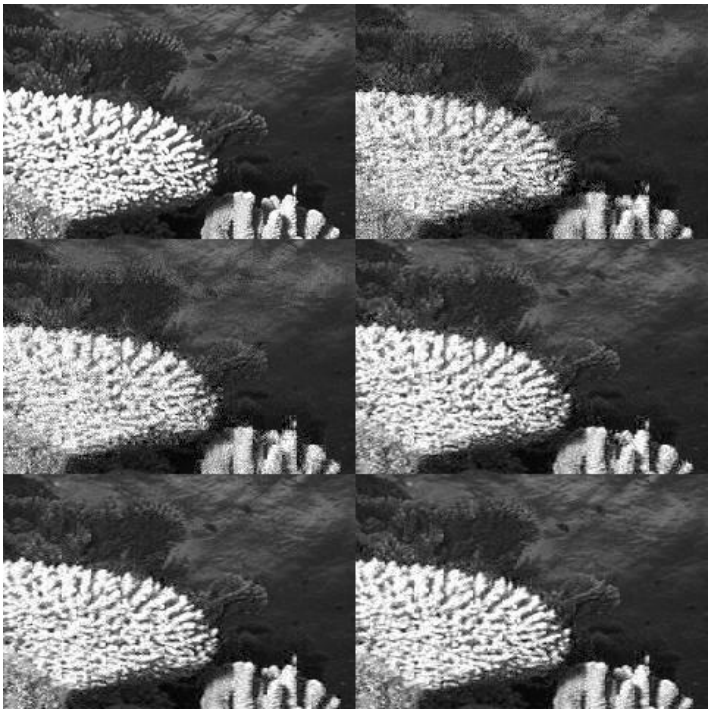

Figure 27: Measurements = 50/100
PSNR Values from top to bottom 20.3366, 20.9485, 23.4671, 24.4368, 24.4420. The images correspond to actual image, random binary, $l_1 - l_\infty$ approach, coherence, double coherence and triple coherence

Figure 28: Measurements = 60/100
PSNR Values from top to bottom 26.6547, 26.9608, 29.6493, 30.1949, 29.9414. The images correspond to actual image, random binary, l_∞ approach, coherence, double coherence and triple coherence

5.3.3 Effect of modified Basis pursuit

In this section random binary matrices are optimized using double coherence and weighted double coherence maximization. Reconstruction is done using basis pursuit and modified basis pursuit algorithm which is defined in section 4. The average PSNR, RRMSE and SSIM values are plotted. We can see that the reconstruction results are better with the modified basis pursuit approach.

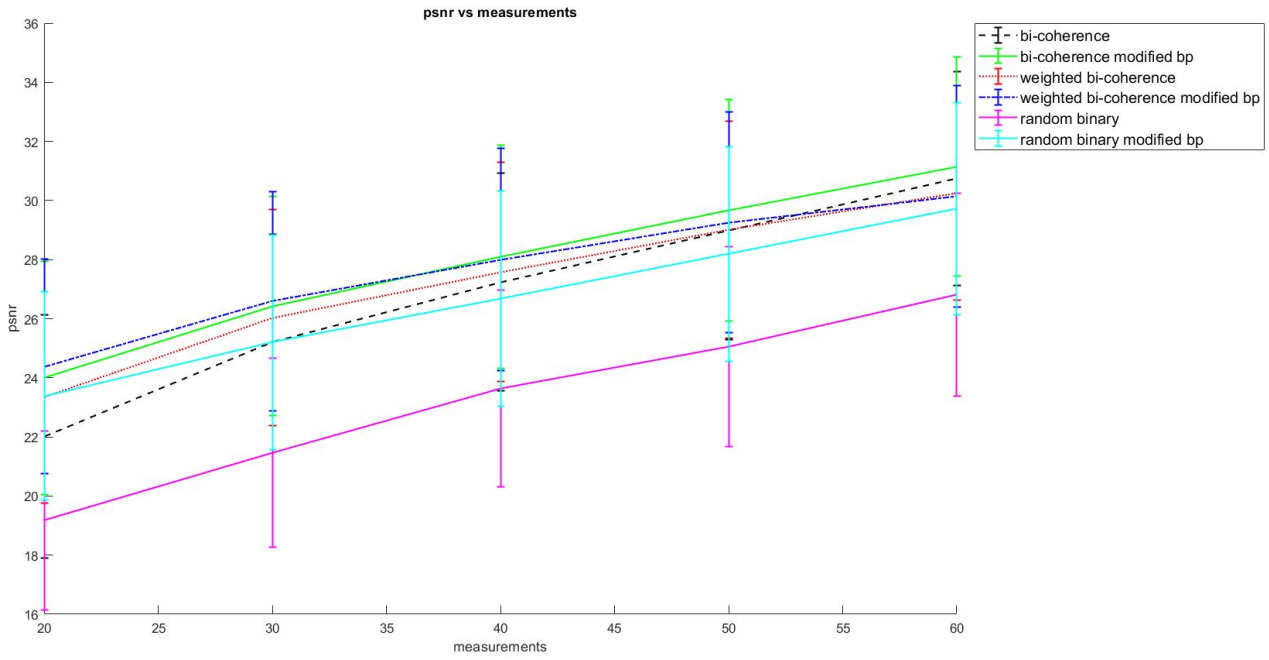

Figure 29: PSNR comparison of modified basis pursuit with basis pursuit on binary matrices

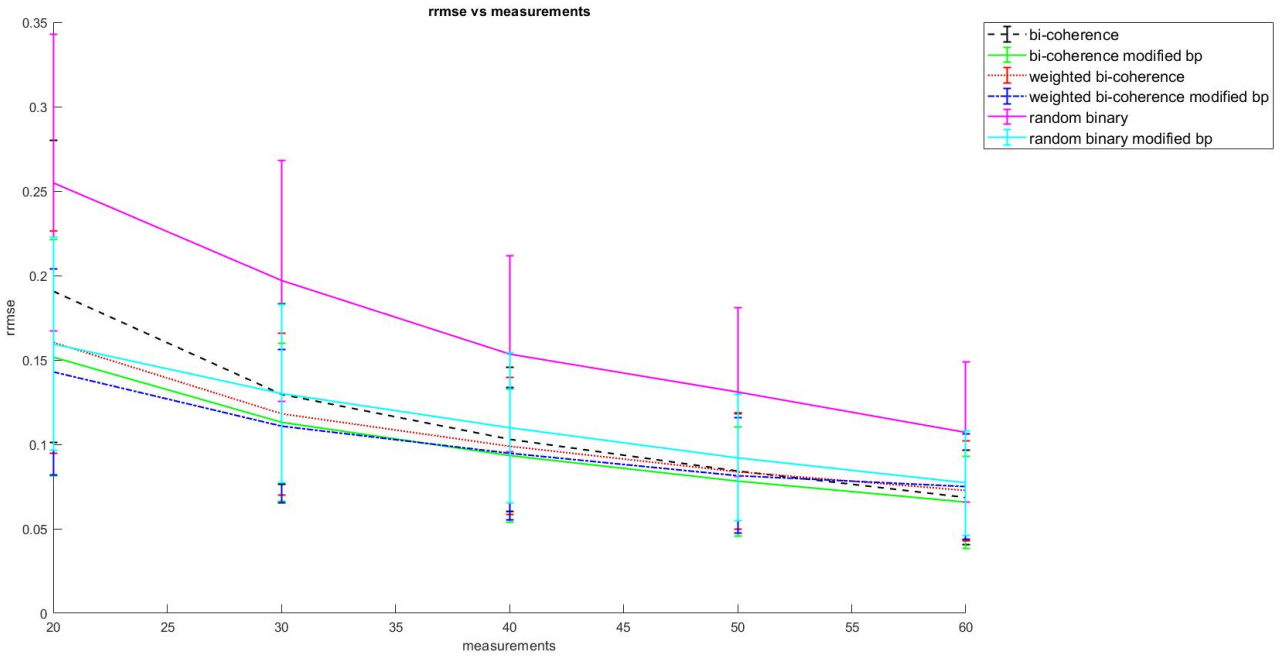

Figure 30: RRMSE comparison of modified basis pursuit with basis pursuit on binary matrices

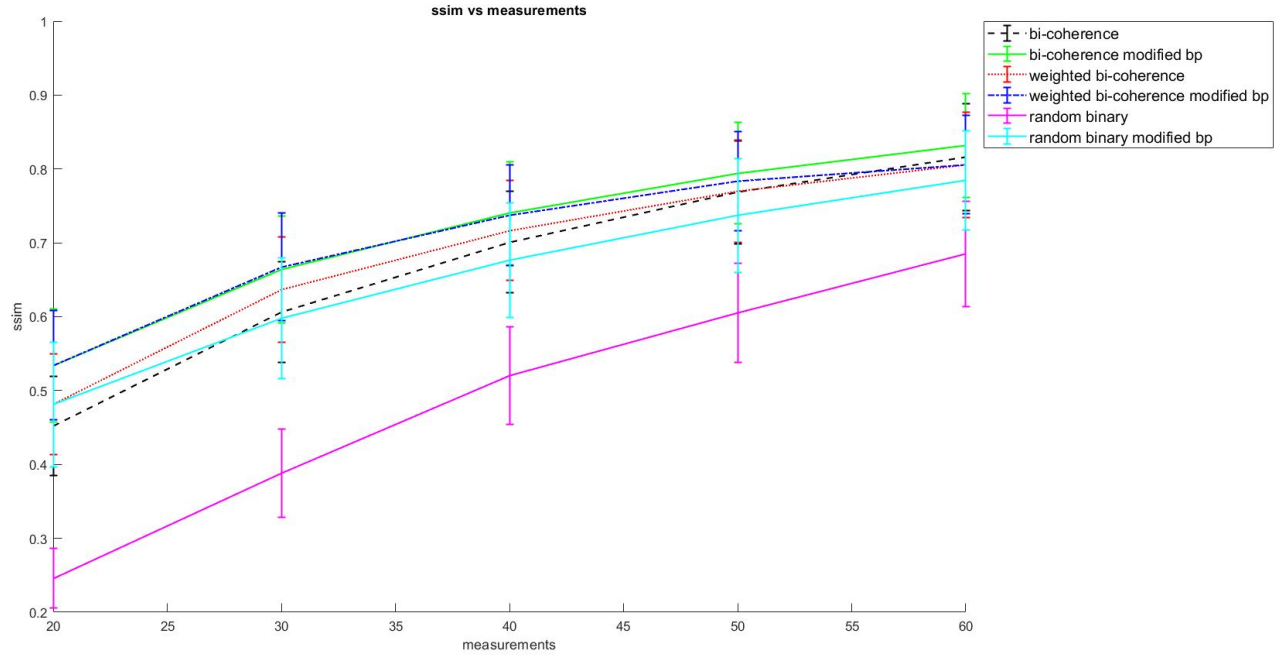

Figure 31: SSIM comparison of modified basis pursuit with basis pursuit on binary matrices

Figure 32: Actual image

Figure 33: Measurements = 20/100
PSNR Values from top to bottom 17.9871, 20.7779, 22.0832, 21.2236, 22.4167, 21.5117. The images correspond to random binary, double coherence, weighted double coherence, random binary with modified bp, double coherence with modified bp and weighted double coherence with modified bp

Figure 34: Actual image

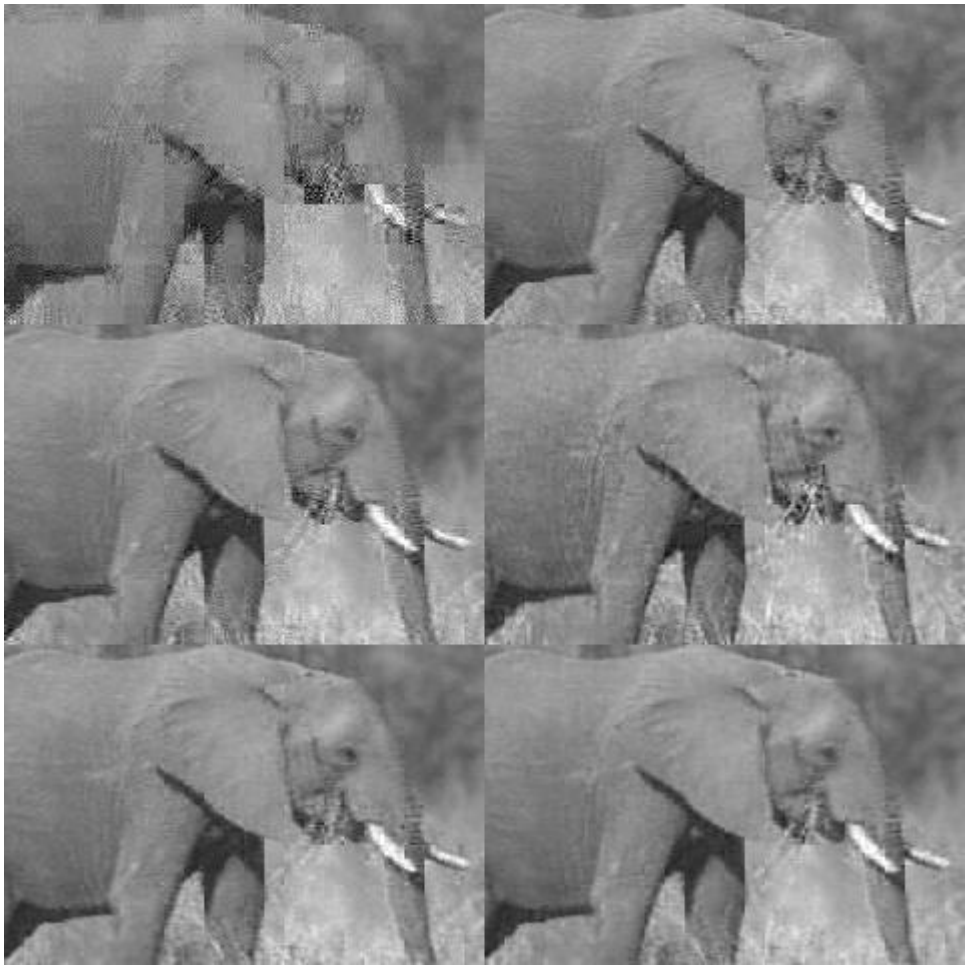

Figure 35: Measurements = 30/100
PSNR Values from top to bottom 23.4629, 27.3765, 28.0298, 27.5039, 28.5242, 28.7598. The images correspond to random binary, double coherence, weighted double coherence, random binary with modified bp, double coherence with modified bp and weighted double coherence with modified bp

Figure 36: Actual image

Figure 37: Measurements = 40/100
PSNR Values from top to bottom 24.7589, 28.0355, 28.2578, 27.2063, 28.8879, 28.7479. The images correspond to random binary, double coherence, weighted double coherence, random binary with modified bp, double coherence with modified bp and weighted double coherence with modified bp

Figure 38: Actual image

Figure 39: Measurements = 50/100
PSNR Values from top to bottom 25.6671, 30.723, 30.6307, 30.5579, 31.7747, 31.1817. The images correspond to random binary, double coherence, weighted double coherence, random binary with modified bp, double coherence with modified bp and weighted double coherence with modified bp

Figure 40: Actual image

Figure 41: Measurements = 60/100
PSNR Values from top to bottom 24.6077, 29.1593, 28.4143, 27.6182, 29.5453, 27.9932. The images correspond to random binary, double coherence, weighted double coherence, random binary with modified bp, double coherence with modified bp and weighted double coherence with modified bp

6 Conclusion and Future Work

In this work, we have proposed some state of the art sensing matrix design ideas using coherence approaches which have shown to provide much improved results. These ideas have also been extended to the case where training data is available. The proposed techniques can be used to design binary sensing matrices which can produce better image reconstruction results.

In the future, we plan to provide some theoretical guarantees on the proposed sensing matrix designs. We also plan to test our designed sensing matrices on more sophisticated state of the art compressive recovery algorithms such as sparse Bayesian learning. Finally, our work on binary matrices can be applied to enhance the work in group testing [11].

References

[1] E. J. Candès, J. K. Romberg, and T. Tao, "Stable signal recovery from incomplete and inaccurate measurements," *Communications on Pure and Applied Mathematics*, vol. 59, no. 8, pp. 1207–1223, 2006.

[2] G. Tang and A. Nehorai, "Computable performance bounds on sparse recovery," *IEEE Transactions on Signal Processing*, vol. 63, no. 1, pp. 132–141, 2015.

[3] E. Candes and T. Tao, "Decoding by linear programming," *IEEE Transactions on Information Theory*, vol. 51, no. 12, pp. 4203–4215, 2005.

[4] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution," *Communications on Pure and Applied Mathematics*, vol. 59, no. 6, pp. 797–829, 2006.

[5] L. Baldassarre, Y. Li, J. Scarlett, B. Gözcü, I. Bogunovic, and V. Cevher, "Learning-based compressive subsampling," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 809–822, 2016.

[6] B. Li, L. Zhang, T. Kirubarajan, and S. Rajan, "Projection matrix design using prior information in compressive sensing," *Signal Processing*, vol. 135, pp. 36 – 47, 2017.

[7] G. Tang and A. Nehorai, "Semidefinite programming for computable performance bounds on block-sparsity recovery," *IEEE Transactions on Signal Processing*, vol. 64, no. 17, pp. 4455–4468, 2016.

[8] Z. Zhou and J. Yu, "Sparse recovery based on q-ratio constrained minimal singular values," *Signal Processing*, vol. 155, pp. 247–258, 2019.

[9] J. E. Fowler, S. Mun, and E. W. Tramel, "Block-based compressed sensing of images and video," *Found. Trends Signal Process.*, vol. 4, p. 297–416, Apr. 2012.

[10] M. F. Duarte, M. A. Davenport, D. Takhar, J. N. Laska, T. Sun, K. F. Kelly, and R. G. Baraniuk, "Single-pixel imaging via compressive sampling," *IEEE Signal Processing Magazine*, vol. 25, no. 2, pp. 83–91, 2008.

[11] S. Ghosh, A. Rajwade, S. Krishna, N. Gopalkrishnan, T. E. Schaus, A. Chakravarthy, S. Varahan, V. Appu, R. Ramakrishnan, S. Ch, M. Jindal, V. Bhupathi, A. Gupta, A. Jain, R. Agarwal, S. Pathak, M. A. Rehan, S. Consul, Y. Gupta, N. Gupta, P. Agarwal, R. Goyal, V. Sagar, U. Ramakrishnan, S. Krishna, P. Yin, D. Palakodeti, and M. Gopalkrishnan, "Tapestry: A single-round smart pooling technique for covid-19 testing," *medRxiv*, 2020.

[12] E. van den Berg and M. P. Friedlander, "Probing the pareto frontier for basis pursuit solutions," *SIAM Journal on Scientific Computing*, vol. 31, no. 2, pp. 890–912, 2008.

[13] E. van den Berg and M. P. Friedlander, "SPGL1: A solver for large-scale sparse reconstruction," December 2019. https://friedlander.io/spgl1.