Sparse and Low-Rank Matrix Recovery

Xiangyu He, Xiujia Li,

***Abstract*—We study the problem of recovering matrices that are sparse in the frequency domain and low-rank in the spatial domain from partial observations. A successive proximal gradient descent (SPGD) algorithm is proposed, combining soft-thresholding and singular value thresholding. We compare it with a data-driven autoencoder. Experiments show that SPGD achieves accurate recovery and outperforms the autoencoder with small training sets. The code and presentation are available in https://github.com/Li888989/Data-compression-final-project-Sparse-and-Low-Rank-Matrix-Recovery.git.**

I. Introduction

A. Motivation and Background

The research of compressed sensing starts from practical needs. In some cases, only part of the data can be obtained, while the complete data is desirable. For example, in MRI, long scanning time burdens patients; and in digital imaging, high-resolution data requires large storage and bandwidth.

The Nyquist sampling theorem states that to reconstruct a signal without distortion, one must sample it at a rate no less than twice its highest frequency component, $2f_{\text{max}}$. However, it is neither feasible nor efficient in many real-world scenarios.

A more effective approach is to fully use the intrinsic structure of the data, such as the sparsity of signals in certain transform domains or the low-rank property of data matrices, to recover the full information from partial observation. This leads to the idea of compressed sensing. The theory was originally introduced by Candès and colleagues [1], as well as Donoho [2].

B. Sparse Property of the Data

In many cases, real-world signals exhibit sparsity in a specific domain. This means that a large number of their coefficients are either exactly zero or very close to zero. For example, communication signals are often sparse in the Fourier domain. Discrete cosine or wavelet transforms can compress natural images well[3].

Xiangyu He and Xiujia Li are master students at TU Delft, The Netherlands. E-mail addresses: {X.He-15, x.li-85}@student.tudelft.nl.

C. Low-Rank Property of the Data

An $m \times n$ matrix A is low-rank if its rank, $k \equiv \text{rank}(A)$, is far less than m and n.

Low-rank or approximately low-rank matrices are common in data science. They arise in applications such as movie recommendation, text processing, survey analysis, medical information, and genomic data. For example, in video sequences, especially surveillance videos, the background usually stays the same, which makes the data low-rank.

Theoretically, the work of Udell and Townsend (2019)[4] explains why the low rank structure often appears.

II. Sparse Structure and the Sparsifying Objective

A. ℓ_0-Norm Formulation for Sparse Recovery

Many natural signals are not sparse in their original domains (such as time or space), but become sparse in transformed domains such as Fourier, wavelet, DCT, or DFT.

Sparsity refers to the property of a signal or matrix having only a small number of non-zero elements, with the majority being zero or close to zero.

A common way to measure sparsity is through the ℓ_0 norm, which counts the number of non-zero entries in a vector or matrix.

Because of this, when we want to find a sparse solution to a problem, we can formulate the objective using the ℓ_0 norm as part of the cost function. For example, we can formulate the problem in the following way:

$$\min_{\hat{H}} \|\mathcal{F}(\hat{H})\|_0 \quad \text{subject to} \quad \mathcal{P}_\Omega(\hat{H}) = G$$

,where $\mathcal{F}$ denotes 2D-DFT.

This formulation directly enforces sparsity by penalizing the number of non-zero entries in H_{hat}.

B. From ℓ_0 to ℓ_1: A Convex Approach to Sparse Recovery

However, the ℓ_0 norm minimization problem is not a convex problem, which makes it computationally intractable in general — it is NP-hard. To overcome this challenge, a common approach is to relax the problem

by replacing the ℓ_0 norm with the ℓ_1 norm, which is the closest convex surrogate to the ℓ_0 norm.

This relaxation is based on the observation that the ℓ_1 norm still promotes sparsity, while resulting in a convex optimization problem. This leads to the following formulation:

$$\min_{\hat{H}} \|\mathcal{F}(\hat{H})\|_1 \quad \text{subject to} \quad \mathcal{P}_\Omega(\hat{H}) = G$$

Under certain conditions (such as the Restricted Isometry Property), the solution to the ℓ_1-relaxed problem is guaranteed to coincide with the solution to the original ℓ_0 minimization.

C. Pratical Setting

In practical scenarios, signals are more often compressible rather than strictly sparse. That is, their nonzero components are not entirely few, but most values are small and can be well-approximated by a few significant ones.

Moreover, due to the presence of measurement noise, the exact equation $\mathcal{P}_\Omega(\hat{H}) = G$ no longer holds. As a result, the compressed sensing problem is typically reformulated as a **constrained** ℓ_1-minimization problem, known as Quadratically Constrained Basis Pursuit:

$$\min_{\hat{H}} \|\mathcal{F}(\hat{H})\|_1 \quad \text{subject to} \quad \|\mathcal{P}_\Omega(\hat{H}) - G\|_2 \leq \epsilon$$

This formulation relaxes the strict equality by allowing a certain level of error, thus improving the model's robustness.

Finally, to facilitate optimization, we often adopt the unconstrained formulation, known as *LASSO* (Least Absolute Shrinkage and Selection Operator):

$$\min_{\hat{H}} \|\mathcal{P}_\Omega(\hat{H}) - G\|_2^2 + \lambda \|\mathcal{F}(\hat{H})\|_1$$

This is a standard convex optimization problem, which can be solved using numerical methods such as gradient descent. If λ is small, we focus more on matching the data; if λ is large, we focus more on making the result sparse.

D. Prox Operation

The objective function is convex, but the ℓ_1 term is not differentiable, so we cannot directly apply gradient descent.

Subgradient method is one choice. However, this method suffers from slow convergence. To improve it, we use proximal gradient descent instead.

With this structure, we can update $\hat{H}$ using the gradient of the smooth part, and handle the non-smooth part with a proximal operation.

For the ℓ_1 norm, the proximal operator has a closed-form solution, known as the soft thresholding function:

$$u_{\text{opt}}[i] = \begin{cases} w[i] - \eta\lambda, & \text{if } w[i] > \eta\lambda \\ w[i] + \eta\lambda, & \text{if } w[i] < -\eta\lambda \\ 0, & \text{otherwise} \end{cases}$$

This operation shrinks small values to 0 and pulls larger values closer to 0, thus encouraging sparsity.

In addition to ℓ_1-based optimization methods, there are other widely used approaches for sparse signal recovery, such as greedy algorithms and data-driven methods.

III. Low-rank Structure and Nuclear Norm Minimization

Considering a low-rank matrix $\mathbf{H} \in \mathbb{R}^{N \times N}$ having rank $R \ll N$, we firstly start with low-rank matrix approximation problem (given full matrix $\mathbf{H}$), then move on to low-rank matrix recovery (given observations $\mathbf{y} = \mathcal{P}_\Omega(\mathbf{H})$).

A. Low-Rank Approximation

The goal is to find the best low-rank approximation to a given matrix $\mathbf{H}$, i.e., to find a matrix of rank R such that it minimizes the difference from $\mathbf{H}$ (smallest squared standard Frobenius norm), and the problem can be formulated as:

$$\min_{\hat{\mathbf{H}}} \left\| \hat{\mathbf{H}} - \mathbf{H} \right\|_F^2 \quad \text{subject to} \quad \text{rank}(\hat{\mathbf{H}}) = R,$$

Although the problem is nonconvex since constraint $\text{rank}(\hat{\mathbf{H}}) = R$ is nonconvex, we can get an explicit solution through simple SVD truncation, the optimal solution is the closest rank-R projection, due to properties such as the rotational invariance of the Frobenius norm, full observations, and the objective being merely to minimize the distance rather than to recover an unknown structure. Those properties together satisfy the requirements of the Eckart–Young–Mirsky Theorem in [5], which guarantees that: Under the Frobenius norm, the optimal rank-R approximation is obtained by truncating the SVD to the first R components.

B. Low-Rank Recovery through Nuclear Norm Minimization

We are only given few observations of $\mathbf{H}$ i.e. $\mathbf{y} = \mathcal{P}_\Omega(\mathbf{H})$. The goal is to find the best low-rank recovery to the given matrix $\mathbf{H}$. The problem can be formulated as:

$$\min_{\hat{\mathbf{H}}} \left\| \mathcal{P}_\Omega(\hat{\mathbf{H}}) - \mathbf{y} \right\|_2^2 \quad \text{subject to} \quad \text{rank}(\hat{\mathbf{H}}) = R, \mathbf{y} = \mathcal{P}_\Omega(\mathbf{H})$$

Unfortunately, this problem can not be solved using SVD truction like matrix approximation in II-B since

only indirect observations are given. The rank constraint leads to a non-convex combinatorial optimization problem (NP-hard), and searching over all matrices of rank R results in a combinatorial explosion.

Hence, convex relaxation is introduced to replace the nonconvex rank function with the convex nuclear norm. Because the nuclear norm is the convex envelope of the rank function over the unit ball of matrices (or under certain constraints such as Frobenius norm or linear observations). This means that minimizing the nuclear norm serves as the best convex relaxation of minimizing rank, and under conditions like incoherence or the Restricted Isometry Property (RIP), the solutions to nuclear norm minimization and rank minimization are provably equivalent. This result was rigorously proven in [6]. The resulting Lagrangian relaxation problem becomes:

$$\min_{\hat{\mathbf{H}}} \left\| \mathcal{P}_\Omega(\hat{\mathbf{H}}) - \mathbf{y} \right\|_2^2 + \lambda \left\| \hat{\mathbf{H}} \right\|_* \quad \text{subject to } \mathbf{y} = \mathcal{P}_\Omega(\mathbf{H}) \tag{1}$$

Furthermore, for an $N \times N$ matrix of rank R, although it contains N^2 entries, its low-rank structure means it has only about $(2N-R)R$ degrees of freedom. This implies that the matrix is highly redundant and contains much less actual information than its size suggests. As a result, it is possible to accurately recover the entire matrix from a small number of observed entries under suitable conditions, provided that the following conditions are met: the matrix rank R is sufficiently small; the row and column spaces exhibit low coherence with the standard basis; and the observed entries are sampled uniformly at random in sufficient number, which should satify $M \geq C \cdot \mu_0 \cdot N^{6/5} R \log N$. Therefore, nuclear norm minimization can exactly recover the underlying low-rank matrix, as proven in [6].

Then, this unconstrained nuclear norm regularized problem (1) can be solved using a proximal algorithm. The data fidelity term is a smooth and differentiable loss, which can be minimized through a gradient step:

$$\mathbf{P}_{t+1} = \hat{\mathbf{H}}_t - \eta \nabla g(\hat{\mathbf{H}}_t) = \hat{\mathbf{H}}_t - 2\eta \cdot \mathcal{P}_\Omega(\hat{\mathbf{H}}_t - \mathbf{y})$$

The nuclear norm term is a nonsmooth convex regularizer which guarantees low rankness. It can be solved using proximal operator of the nuclear norm, defined as:

$$\begin{aligned} \hat{\mathbf{H}}_{t+1} &= \text{prox}_{\eta h}(\mathbf{P}_{t+1}) \\ &= \arg\min_{\hat{\mathbf{H}}} \left\{ \frac{1}{2\eta} \left\| \hat{\mathbf{H}} - \mathbf{P}_{t+1} \right\|_F^2 + \lambda \left\| \hat{\mathbf{H}} \right\|_* \right\} \end{aligned}$$

This proximal operator has a closed-form solution via singular value thresholding (SVT). Let $\mathbf{P}_{t+1} = U\Sigma V^\top$, where $\Sigma = \text{diag}(\sigma_1, \ldots, \sigma_r)$, the proximal update is:

$$\hat{\mathbf{H}}_{t+1} = U \cdot \text{diag}\left(\max(\sigma_i - \eta\lambda, 0)\right) \cdot V^\top$$

i.e. $\text{prox}_{\eta h}(\mathbf{P}) = \mathcal{S}_{\eta\lambda}(\mathbf{P})$

Furthermore, there are other algorithms to recover low rank matrix other than nuclear norm method, such as Iterative Hard Thresholding (IHT), which maintains an explicit rank constraint by projecting onto low-rank matrices after each gradient step. They are nonconvex but often offer computational advantages for large scale problems.

IV. Problem Formulation and Methodologies

A. Successive Proximal Gradient Operations

Based on the convex optimization algorithms presented in Section II and Section III, the sparse and low-rank matrix recovery problem can be solved using successive proximal gradient descent(SPGD) operations, which iteratively minimize a composite objective function containing a data consistency term, an l_1-norm in 2D-DFT domain inducing sparsity and a nuclear norm promoting low-rankness. The problem can be formulated as:

$$\begin{aligned} \min_{\hat{\mathbf{H}}} \quad & \left\| \mathcal{P}_\Omega(\hat{\mathbf{H}}) - \mathbf{y} \right\|_2^2 + \lambda_1 \left\| \mathbf{X} \right\|_1 + \lambda_2 \left\| \hat{\mathbf{H}} \right\|_* \\ \text{subject to} \quad & \hat{\mathbf{H}} = \mathbf{X} \end{aligned}$$

where $\mathcal{P}_\Omega$ is a mask matrix containing zeros and ones for sampling, $\mathcal{F}$ denotes 2D-DFT, $\mathbf{X} = \mathcal{F}(\hat{\mathbf{H}}) = \mathbf{U_N} \hat{\mathbf{H}} \mathbf{U_N^H}$, and $\mathcal{F}^{-1}$ denotes inverse 2D-DFT, $\hat{\mathbf{H}} = \mathcal{F}^{-1}(\mathbf{X}) = \mathbf{U_N^H} \mathbf{X} \mathbf{U_N}$.

At each iteration, the update process contains 3 steps:
Step 1 (gradient step): Perform gradient descent on the first term to ensure the reconstruction is close to the measurement under the mask matrix.

$$\mathbf{P}_{t+1} = \hat{\mathbf{H}}_t - \eta \nabla g(\hat{\mathbf{H}}_t) = \hat{\mathbf{H}}_t - 2\eta \cdot \mathcal{P}_\Omega(\hat{\mathbf{H}}_t - \mathbf{y})$$

Step 2 (proximal operation): The intermediate result $\mathbf{P}_{t+1}$ is transformed into the 2D-DFT domain, and is treated using element-wise soft thresholding to enforce sparsity as explained in Section II.

$$\mathbf{Q}_{t+1} = \mathcal{F}^{-1}\left(\mathrm{S}_{\eta\lambda_1}\left(\mathcal{F}(\mathbf{P}_{t+1})\right)\right)$$

where $\mathrm{S}_{\eta\lambda_1}(X) = \text{sign}(X) \cdot \max(|X| - \eta\lambda_1,\, 0)$

Step 3 (proximal operation): Apply singular value thresholding which promotes low-rankness to update $\hat{\mathbf{H}}$ as explained in Section III.

$$\hat{\mathbf{H}}_{t+1} = \mathcal{S}_{\eta\lambda_2}(\mathbf{Q}_{t+1}) = \mathbf{U} \cdot \text{diag}\left(\max(\sigma_i - \eta\lambda_2, 0)\right) \cdot \mathbf{V}^\top$$

where $\mathbf{Q}_{t+1} = \mathbf{U}\Sigma\mathbf{V}^\top$

B. Autoencoder

In this project, the goal is to recover the full matrix H from its subsampled observations $y = \mathcal{P}_\Omega(H)$. The matrix H is low-rank and sparse in the 2D DFT domain. This structure matches well with the strengths of an autoencoder. By learning shared compression patterns from enough training samples, the autoencoder can take subsampled measurements as input and reconstruct the full matrix. Fig. 1 shows the structure of autoencoder used in our project.

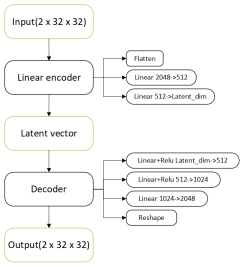


Fig. 1: Structure of Autoencoder in this Peoject

V. Results

We are given a dataset containing 1000 matrices H of size 32×32. These matrices are low-rank. Moreover, they exhibit sparsity in the 2D Discrete Fourier Transform (2D DFT) domain.

A. Hyperparameter tuning in successive PGD

We observed that if the learning rate is chosen properly, the error becomes stable after about 100 iterations. So, we set the following three hyperparameters based on this idea.

Learning rate in gradient step: The learning rate is chosen so that the error can become stable in about 100 iterations. If the learning rate is too large, the error will go up and not converge. If it is too small, the algorithm will converge very slowly.

λ_1 **Hyperparameter in sparse proximal operation:** In the frequency domain, sparsity is controlled using soft-thresholding. The parameter λ_1 controls the balance between keeping useful information and removing noise. To choose a suitable λ_1, we manually check the number of non-zero entries before and after thresholding. The goal is to adjust λ_1 so that, after about 100 iterations, the number of non-zero entries in the frequency-domain matrix is close to that of the original matrix. It should be neither too many nor too few. This helps ensure a good recovery result.

λ_2 **Hyperparameter in low-rank proximal operation:**The parameter λ_2 controls the strength of soft-thresholding on singular values. It directly affects the rank of the estimated matrix $\hat{H}$. We choose λ_2 by checking how many singular values are above the threshold before and after the operation. The goal is to keep a number of singular values close to the true rank after 100 iterations. This helps reduce noise while preserving the main structure.

B. Results of successive PGD

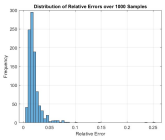


Fig. 2: Distribution of Relative Errors over 1000 Samples

Fig. 2 shows how the relative reconstruction errors are distributed across all 1000 samples. As we can see, most of the errors are very small—under 0.02—which means the method works well in most cases, even when only 10% of the data is used. For example, the 23rd sample has a relative error of 0.0233, which is slightly higher but still acceptable.

The example below, which is used for visual comparison in Fig. 3 and Fig. 4, has a relative error of 0.0233. Although slightly above the average, it still reflects a successful reconstruction in both time and frequency domains.

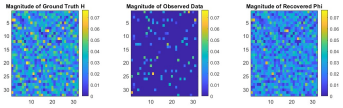


Fig. 3: Comparison of Magnitude: Original vs. 10% Sampled vs. Recovered

Fig. 3 compares the magnitudes of three matrices in the time domain: the original one (left), the version with only 10% of its entries observed (middle), and the recovered result (right). The recovered matrix looks very similar to the original, both in structure and in value, which shows that the method successfully fills in the missing information.

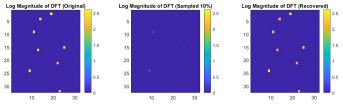
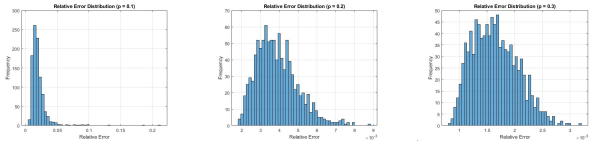


Fig. 4: Comparison of DFT Log Magnitude: Original vs. 10% Sampled vs. Recovered

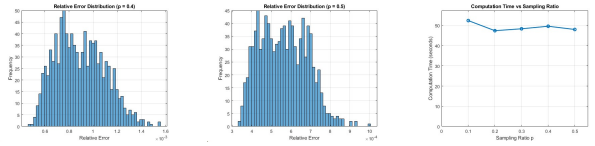
Fig. 4 shows the log-magnitude of the 2D DFTs for the same three cases. The original and recovered frequency patterns are quite similar, while the sampled one looks

much more distorted. This suggests that the algorithm is able to reconstruct the key frequency components and maintain sparsity in the frequency domain.

C. Discussions about PGD



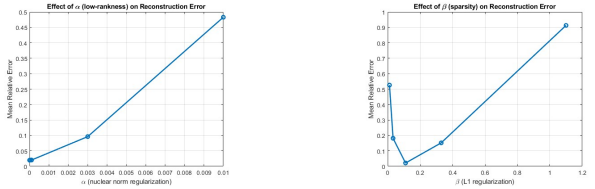
(a) $P = 0.1$ (b) $P = 0.2$ (c) $P = 0.3$



(d) $P = 0.4$ (e) $P = 0.5$ (f) Running Time

Fig. 5: Reconstruction error and runtime analysis.

As the sampling rate increases, the reconstruction error clearly decreases, while the running time remains nearly constant.



(a) Parameter in low rankness (b) Parameter in sparsity

Fig. 6: Balance between promoting sparsity and low rankness

We adjust the magnitude of the parameters related to low-rankness and sparsity based on a manually tuned set of values. Fig. 6 shows how the reconstruction error varies with different combinations. The lowest point is the manually tuned set.

D. Results of autoencoder

Since the feedforward neural network requires sufficient data for effective training, but we only have 1000 matrices, 20% of which must be reserved for testing, the performance is understandably limited. During training, we observed that the model never truly captures the sparsity in the 2D DFT domain or the low-rank property of the matrices. Although the training error decreases, the validation error remains around 1, indicating poor generalization.

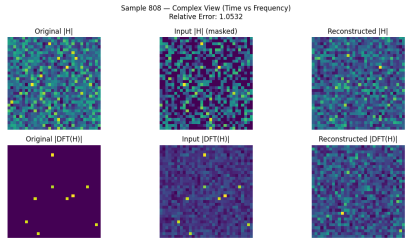


Fig. 7: Comparison of DFT Log Magnitude: Original vs. 60% Sampled vs. Recovered

The reconstruction results are clearly poor, as seen in Fig. 7. This is mainly due to the limited training data. If we had 100,000 matrices instead of just 1,000, the neural network could learn much more effectively.

VI. Discussion

In this project, we implemented two different approaches for matrix reconstruction: a classical iterative algorithm (Successive PGD) and a data-driven method (Autoencoder).

The Successive PGD method consistently achieves low relative reconstruction errors across the 1000 test samples. In contrast, the autoencoder performs significantly worse, which is due to limited data set.

The successive PGD method offers several advantages. It requires much smaller training data compared with autoencoders, and can directly exploit known structural priors, such as sparsity and low-ranknes. However, one notable drawback is the need for manual tuning of hyperparameters.

The autoencoder-based approach enables fast reconstruction once trained and has the capacity to learn complex nonlinear structures from data. However, its performance heavily relies on the availability of large and diverse training datasets. It is also sensitive to the choice of network architecture and training configuration. Unlike PGD, it does not inherently exploit known physical priors unless they are explicitly incorporated into the model design.

Overall, in a setting with limited data set like ours, classical model-based methods such as PGD clearly outperform data-driven approaches. However, with access to a much larger dataset (e.g., 100,000 matrices), the autoencoder may be able to match or even surpass PGD, provided it learns the relevant structures effectively.

References

[1] E. J. Candès, J. Romberg, and T. Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," *IEEE Transactions on information theory*, vol. 52, no. 2, pp. 489–509, 2006.

[2] D. L. Donoho, "Compressed sensing," *IEEE Transactions on information theory*, vol. 52, no. 4, pp. 1289–1306, 2006.

[3] S. Mallat, *A wavelet tour of signal processing*. Elsevier, 1999.

[4] M. Udell and A. Townsend, "Why are big data matrices approximately low rank?" *SIAM Journal on Mathematics of Data Science*, vol. 1, no. 1, pp. 144–160, 2019.
[5] M. A. Davenport and J. Romberg, "An overview of low-rank matrix recovery from incomplete observations," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 4, pp. 608–622, 2016.
[6] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational Mathematics*, vol. 9, no. 6, pp. 717–772, 2009.