Hybrid-(L_1, L_2) Sampling for Elementwise Matrix Sparsification

Final Project based on Kundu et al.,2015

Fernando Spadea Yogesh Agrawal

Machine Learning and Optimization
Group G3

Outline

About the Paper

Title:

Elementwise Matrix Sparsifiacation and Reconstruction

Authors:

Abhisek Kundu, Rensselear Polytechnic Institute

Published: Thesis for the degree of DOCTOR OF PHILOSOPHY, Computer Science Department, RPI, 2015

Hyperlink: https://www.cs.rpi.edu/ magdon/LFDlabpublic.html/Theses/kundu2015/Thesis_Kundu_Abhisek.pdf

Brief History

1. Element-wise sparsification initially pioneered by Achlioptas (2001); Achlioptas & McSherry (2007) to be done by $L2$ sampling technique
2. Achlioptas (2001); Achlioptas & McSherry (2007) observe small entries need to sampled using $L1$
3. Drineas & Zouzias (2011) Bypass $L1$ by zeroing out small entries. Gives better result and rise to Matrix-Bernstein inequality Recht (2011).
4. Achlioptas et al. (2013) Show $L1$ sampling in isolation better than previous method.
5. Kundu (2015) Our author provide algorithm to retain good properties of $L2$ while regularizing smaller entries using $L1$.

Introduction to the paper

Aim: To present an algorithm to perform Element wise sparsification using sampling probabilities that depend on both: squares and absolute values of entries.

Main contribution: Propose sampling probabilities of the form:

$$p_{ij} = \alpha \frac{|\mathbf{A}_{ij}|}{||\mathbf{A}||_1} + (1-\alpha)\frac{\mathbf{A}_{ij}^2}{||\mathbf{A}||_F^2} \quad \text{where} \quad \alpha \in (0,1] \tag{1}$$

Notation:

1. $\mathbf{X}_{ij}$ is matrix with i-rows and j=columns, $\mathbb{E}(\mathbf{X})$ is the expectation of $\mathbf{X}$
2. $||\mathbf{X}||_F$, $||\mathbf{X}||_1$ and $||\mathbf{X}||_2$ refer to the Frobenious norm, 1-norm and 2-norm(spectral norm) of the matrix $\mathbf{X}$ respectively.
3. $\tilde{\mathbf{X}}$ refers to sparsified form of matrix $\mathbf{X}$
4. σ refers to singular values
5. ϵ is accuracy parameter, s is number of sparsed elements.
6. Ω is multi-set of sampled indices

Introduction to hybrid sampling technique

Goal: Combines the benefits of L1 sampling and L2 sampling to achieve a more efficient and accurate sparsification process.

Definitions:

- *Matrix Sparsification*: Process of reducing the number of non-zero elements in a matrix while preserving its essential properties, such as spectral or structural information.
- *Element-wise sparsification of a matrix using L1 sampling*Technique used to create a sparse representation of a matrix by retaining only a small fraction of its original elements. The probability of each element to be selected is given by the ratio of the absolute value of that element and the l_1 norm of the matrix.
  Good at preserving structure of input matrix.
- *Element-wise sparsification of a matrix using L2 sampling* The probability of each element to be selected is given by the ratio of the ratio of squared value of that element and the Frobenius norm of the matrix.
  Good at preserving Spectral properties of matrix.

Spectral vs Structural Properties

- *Spectral Properties* Provide insight into Matrix's behaviour especially under linear transformations.
  Examples of Spectral Properties: Eigenvalues and eigenvectos, Singular values, Spectral radius, condition number, etc.
- *Structural Properties*: Related to shape, pattern of non-zero elements. Can be used to analyze features that impact efficiency of algorithms and storage requirements.
  Example of Structural Properties: Symmetry, Diagonal Dominance, Block Structure, etc.

Motivation

Motivation for Hybrid Sampling:
Assigns proabability to each element based on weighted combination of their L_1 and L_2 norms, balancing the preserverance of structural and spectral properties.

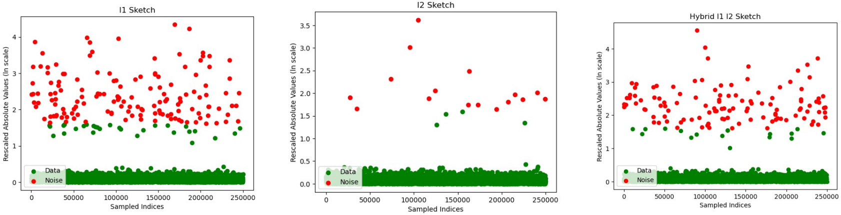


Figure: Elements of sparse Sketches $\tilde{\mathbf{A}}$ produced by three sampling techniques as instructed by the author. The y-axis plots rescaled absolute values (in $\ln$ scale) of $\tilde{\mathbf{A}}$ corresponding to sampled indices. L_1 produces elements with controlled variance but mostly samples noise, L_2 samples a lot of data although producing large variance of rescaled elements. Hybrid uses L_1 as regularizer while sampling fairly large data elements.

Advantages and Disadvantages of Hybrid Sampling

Advantages:

1. Balanced sparsification
2. Flexibility
3. Improved accuracy
4. Scalability

Possible Disadvantages:

1. complexity
2. Tuning parameters
3. Loss of sparsity
4. No universal solution

Possible Struggles of Hybrid Sampling

Types of data input or problem charateristics where proposed Hybrid sampling might struggle:

1. Highly imbalanced data
2. Noise-dominated data
3. Strongly structured matrices
4. Highly sparse matrices
5. Data with specific requirements

Example of Proposed struggle

Input Matrix: Large block diagonal matrix composed of smaller, dense submatrices along main diagonal, and off diagonal elements zero. Example: Images of license plate numbers.[1]

Possible reason for inefficiency:

1. The matrix structure is already sparse
2. Loss of block structure
3. Inefficient sampling

Possible alternative: Block-wise sparsification, hierarchical sampling,etc.

[1] Performance of algorithm on data with License plate number images can be seen in experiments and results

Sampling operator and its construction

Let sampling operator be $S_\Omega : \mathbb{R}^{m \times n} \to \mathbb{R}^{m \times n}$ which will extract the elements from our given matrix $\mathbf{A}$.

Equation 1: Sampling Operator

$$S_\Omega = \frac{1}{s} \sum_{t=1}^{s} \frac{\mathbf{A}_{i_t j_t}}{p_{i_t j_t}} \mathbf{e}_{i_t} \mathbf{e}_{j_t}^T \quad \text{where}(i_t, j_t) \in \Omega$$

Construction:

- Find the optimal α^*, the number of non-zero elements s^*, and the probabilities p_{ij}
- The algorithm then samples s^* indices from the input matrix A, with the probability of selecting each element (i, j) being proportional to p_{ij}.
- For each sampled index (i, j),

$$S_\Omega(i, j) \mathrel{+}= \frac{A(i, j)}{p_{ij} s^*} \tag{2}$$

Algorithm

Use Algorithm 1 based on probability distribution calculated in eq. 1

Algorithm 1 Element-wise Matrix Sparsification

Require: $A \in \mathbb{R}^{m \times n}$, accuracy parameter $\epsilon > 0$.

1: Set s as in eq. 9.

2: **for** $t = 1 \ldots s$ (i.i.d. trials with replacement) **do**

3: Randomly sample pairs of indices $(i_t, j_t) \in [m] \times [n]$ with $P[(i_t, j_t) = (i, j)] = p_{ij}$, where p_{ij} are as in eq. 1, using α as in eq. 8.

4: **end for**

5: **Output(sparse)**: $S_\Omega(A) = \frac{1}{s} \sum_{t=1}^{s} \frac{A_{i_t j_t}}{p_{i_t j_t}} e_{i_t} e_{j_t}^T$.

Idea behind Algorithm

Theorem: Lemma of Matrix Bernstein Theorem

[a] Let $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\epsilon > 0$ be an accuracy parameter. Let $\mathbf{S}_\Omega$ be the sampling operator defined above and assume we generate multi-set Ω using the above sampling probabilities. Then, with probability at least $1 - \delta$,

$$||\mathbf{S}_\Omega(\mathbf{A}) - \mathbf{A}||_2 \leq \epsilon ||\mathbf{A}||_2 \tag{3}$$

if

$$s \geq \frac{2}{\epsilon^2 ||\mathbf{A}||_2^2} \left(\rho^2(\alpha) + \gamma(\alpha) \in ||\mathbf{A}||_2/3\right) ln \left(\frac{m+n}{\delta}\right) \tag{4}$$

[a] Mathematical Proof presented in the pdf submitted

Here, Matrix-Bernstein inequality gives functional norm in (4). Lemma of Matrix-Bernstein gives proof for the quality of the approximation produced.

Functions Involved in Matrix Bernstein Inequality

Xi function (ξ):

Computes an intermediate value representing a weighted combination of matrix A's element-wise absolute values and element-wise squares. The weighting is determined by the parameter α.

$$\xi_{ij} = \frac{||\mathbf{A}||_F^2}{\alpha \frac{||\mathbf{A}||_F^2}{|||\mathbf{A}||_{ij}|||\mathbf{A}||_1} + (1-\alpha)} \tag{5}$$

- Represents the trade-off between the absolute value of matrix elements and the squared Frobenius norm of the matrix.
- The parameter alpha is used to control the trade-off between these two factors.
- In the context of the algorithm, ξ is a matrix of the same size as the input matrix $\mathbf{A}$. When alpha is closer to 1, the algorithm prioritizes minimizing the sum of absolute values of the errors, and when alpha is closer to 0, it prioritizes minimizing the sum of squared errors.

Functions Involved in Matrix Bernstein Inequality

Rho squared function (ρ^2)

The ρ^2 function calculates the difference between the maximum row-wise and column-wise sums of ξ and the square of the smallest singular value (σ^2_{min}) of matrix A.

$$\rho^2(\alpha) = \max\left(\max_i \sum_j \xi_{ij}, \max_j \sum_i \xi_{ij}\right) - \sigma^2_{\text{min}} \tag{6}$$

where σ_{min} refers to the smallest singular value of $\mathbf{A}$.

- Computes an error metric that is used to assess the quality of the sparse approximation.
- In particular, it measures the maximum deviation of the row and column sums of ξ from the minimum singular value squared of the input matrix $\mathbf{A}$.
- A lower ρ^2 value indicates a better approximation, as it means that the sparse approximation closely preserves the row and column sums of the input matrix

Functions Involved in Matrix Bernstein Inequality

Gamma function (γ)

γ controls the incoherence of the matrix, i.e, larger values of γ makes data more spiky.

$$\gamma(\alpha) = \max_{i,j} \frac{||\mathbf{A}||_1}{\alpha + (1-\alpha)\frac{|\mathbf{A}_{ij}|||\mathbf{A}||_1}{||\mathbf{A}||_F^2}} + ||\mathbf{A}||_2 \quad (7)$$

- Controls the error bound of the matrix approximation, ensuring that the sparse approximation does not deviate too much from the original matrix.
- In particular, creates an upper bound on the errors of our approximation.
- Goal of the algorithm is to minimize the error bound (given by γ) while creating the sparse representation of the matrix.

Optimal α and s

To find the optimal α^* and s^* we can solve following optimization problem:[2]

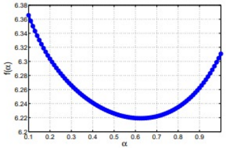
$$\alpha^* = \min_{\alpha\in(0,1]} f(\alpha) = \rho^2(\alpha) + \gamma(\alpha)\epsilon||\mathbf{A}||_2/3 \tag{8}$$

$$s^* = \frac{2}{\epsilon^2||\mathbf{A}||_2^2}\left(\rho^2(\alpha^*) + \gamma(\alpha^*) \in ||\mathbf{A}||_2/3\right) ln\left(\frac{m+n}{\delta}\right) \tag{9}$$

[2] From the Matrix Bernstein Inequality

Verification of value of optimal alpha

Author's verification of optimal alpha not being at extreme ends.



Plot of $f(\alpha)$ in eqn (1.6) for data $A_{0.1}$. We use $\epsilon = 0.05$ and $\delta = 0.1$. x-axis plots α and y-axis is in $\log_{10}$ scale. For this data, $\alpha^* \approx 0.62$.

Figure: For synthetic data of $500 * 500$ perturbed by noise of standard deviation 0.1, optimal value of alpha is at 0.62

Verification of value of optimal alpha

Our experiments on different sizes and noise level to verify authors claim:

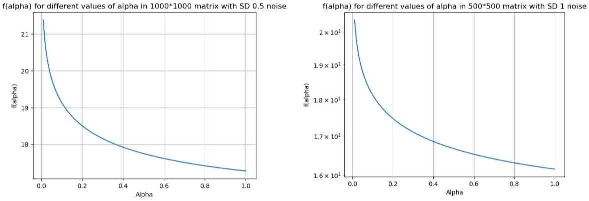


Figure: Plots of f(α)

for perturbed matix of size $1000 * 1000$ with noise 0.5 and size $500 * 500$ with noise 1, x-axis plots α and y-axis plots $f(\alpha)$ on log scale.

We find that optimal value of alpha occurs near 1 showing that l_1 sampling on its own would be sufficient.

Experiment Description

- Tested matrix sparsification algorithm on matrices constructed from different datasets using different parameters:
  - sampling size s
    - $s = sMult * k * (n + m)$
    - where k is the matrix rank
    - where n and m are the dimensions of the matrix
    - where $sMult$ will be set to 1, 3, and 5
  - l_1 and l_2 weight value α
    - α is set to values 0 to 1 with step size 0.1
    - $\alpha = 1$ is just l_1 sparsification and $\alpha = 0$ is just l_2 sparsification
- We measure our error for our sparsified matrix, $\tilde{A}$, against our original matrix, A, using the following equation: $Error = \log_2(||A - \tilde{A}||_2/||A||_2)$
- We then plot our errors against values of α for different values of s.

1, 6, and 9 Handwritten Digits

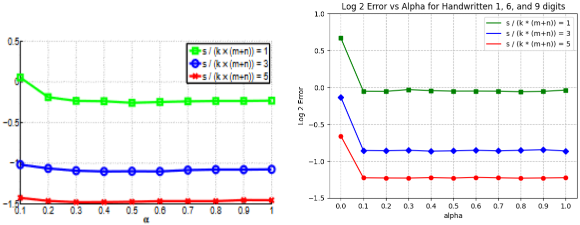


Figure: Our results for the sparsification algorithm against all the 1, 6, and 9 digits in the MNIST Handwritten Dataset on the right compared to the original author's results for the same digits from a different dataset on the left.

- Our results show a similar pattern, although they are a bit worse.
    - Discrepancy due to using different datasets with different resolutions (28x28 vs 18x18)

General 2000 Handwritten Digits

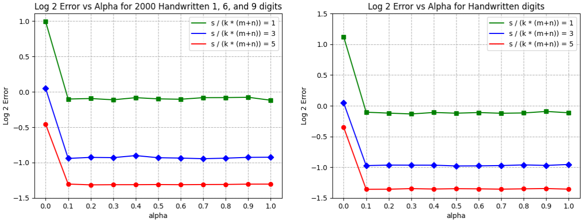


Figure: Our results for testing the algorithm on 2000 Handwritten digits. On the left, only digits 1, 6, and 9 were included, and the right includes all digits.

- The results were surprisingly better when we allowed any kind of digit to be included.
- However, hybrid sparsification did not have an advantage over l_1 sparsification.

License Plate Images 1 and 4

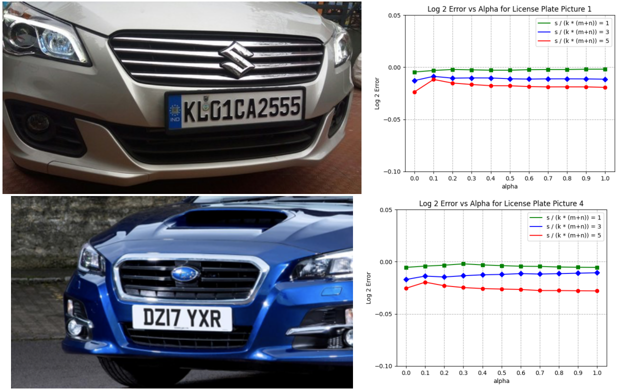


Figure: Our Results for the first and fourth images of the license plate dataset.

- Very poor results, likely due to the block matrix nature of these images
- Even l_2 sampling does better than hybrid sampling, and it even performs better than l_1 in some cases.

License Plate Images 2 and 3

Figure: Our Results for the second and third images of the license plate dataset.

- Also had very poor results, likely for the same reasons
- Oddly, matrix sparsification performs worse as sMult reaches 5.
- Either way, hybrid sparsification does not out-perform pure l_1 and l_2 sampling.

Football Scores

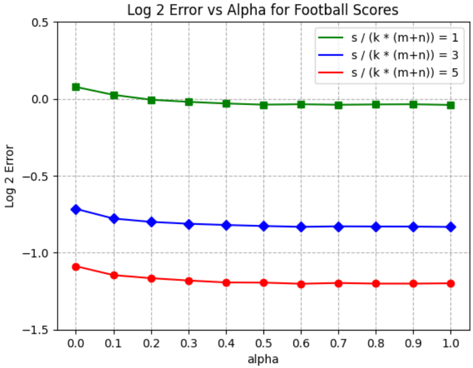


Figure: Our results on a dataset of home and away scores for 13516 games.

- Decent results overall, but hybrid sparsification does not outperform l_1 sparsification.

Conclusion

Overall, our results weren't very impressive. We were able to replicate the author's findings for handwritten digits but We found that hybrid sampling didn't noticeably improve our results with any of the tested datasets, and was actually worse in certain cases. This isn't to say that author's hybrid sparsification algorithm is worthless, seeing as he found cases where it performed well in comparison to pure l_1 and l_2 sampling, but it isn't as widely applicable as one may hope. If computation power or time is not a concern, then there is no harm in performing hybrid sparsification as it will result in the optimal combination of l_1 and l_2 sampling for your use case. However, if computation is a concern, one may be better off just trying out l_1 Achlioptas et al. (2013) or l_2 sampling Drineas & Zouzias (2011) as the benefits of hybrid sampling can often be minimal or nonexistent.

Going Forward

One major advantage of hybrid sampling presented by the author is the fast approximation of PCA of the matrix using PCA of sparse matrix produced by one-pass hybrid sampling. Going forward, we would like to study the algorithms he performs to modify the hybrid sampling to one-pass hybrid sampling and then study his findings regarding the PCA approximation. Then only we can truly discuss the true need of hybrid sampling as stated by the author.

Thank you!

D Achlioptas. Fast computation of lowrank matrix approximation. In *Proceedings of the thirty-P third annual ACM symposium on Theory of computing*, pp. 611–618. ACM, 2001.

Dimitris Achlioptas and Frank McSherry. Fast computation of low-rank matrix approximations. *Journal of the ACM (JACM)*, 54(2):9–es, 2007.

Dimitris Achlioptas, Zohar Karnin, and Edo Liberty. Matrix entry-wise sampling: Simple is best. *Submitted to KDD*, 1:1–4, 2013.

Petros Drineas and Anastasios Zouzias. A note on element-wise matrix sparsification via a matrix-valued bernstein inequality. *Information Processing Letters*, 111(8):385–389, 2011.

Abhisek Kundu. *Element-wise matrix sparsification and reconstruction*. Rensselaer Polytechnic Institute, 2015.

Benjamin Recht. A simpler approach to matrix completion. *Journal of Machine Learning Research*, 12(12), 2011.