antGLasso: An Efficient Tensor Graphical Lasso Algorithm

Anonymous Author(s)
Affiliation
Address
email

Abstract

The class of bigraphical lasso algorithms (and, more broadly, 'tensor'-graphical lasso algorithms) has been used to estimate dependency structures within matrix and tensor data. However, all current methods to do so take prohibitively long on modestly sized datasets. We present a novel tensor-graphical lasso algorithm that directly estimates the dependency structure, unlike its iterative predecessors. This provides a speedup of multiple orders of magnitude, allowing this class of algorithms to be used on large, real-world datasets.

1 Introduction

We often make independence assumptions to simplify the modelling of complex data. The strength of an independence assumption exists on a spectrum. One could assume complete independence, independence except on a sparse subset, or independence of samples but not features, among others.

For conditional dependencies, there is a natural way to model Gaussian data. Two datapoints x, y are *conditionally independent* (with respect to a dataset $\mathcal{D}$) if knowing one provides no information about the other that is not already contained in the rest of the dataset: $\mathbb{P}(x|y, \mathcal{D}) = \mathbb{P}(x|\mathcal{D})$. For normally distributed data, conditional dependencies are encoded in the inverse of the covariance matrix (the 'precision' matrix). Two datapoints are conditionally dependent on each other if and only if their corresponding element in the precision matrix is zero. If our dataset $\mathbf{d}$ were in the form of a vector, we could then model it as $\mathbf{d} \sim \mathcal{N}(\mathbf{0}, \Psi^{-1})$ for precision matrix Ψ.

However, datasets are often not simple vectors - for example, single-cell RNA sequencing data (scRNA-seq) comes in the form of a matrix of gene expression counts whose rows are cells and columns are genes. Video data naturally requires a third-order tensor of pixels to represent it - rows, columns, and frames. To address this, we 'vectorize' matrices by stacking the columns vertically. Vectorization of tensors proceeds in much the same way.

Our matrix-variate dataset $\mathcal{D}$ can then be represented as $\mathrm{vec}[\mathcal{D}] \sim \mathcal{N}(\mathbf{0}, \Psi^{-1})$. Suppose our matrix were $n \times p$ dimensional - then $\mathrm{vec}[\mathcal{D}]$ has np elements, and its precision matrix has n^2p^2 elements. This poses a problem: the estimation of our precision matrix requires substantially more parameters than we have in our dataset. Since the columns of the matrix have interdependencies and the rows of the matrix have interdependencies, to reduce space we can compute the interdependencies of the elements of the matrix via some deterministic function of the row-wise and column-wise dependencies. That is, for some function ζ, $\mathrm{vec}[\mathbf{D}] \sim \mathcal{N}(\mathbf{0}, \zeta(\Psi_{\mathrm{row}}, \Psi_{\mathrm{col}})^{-1})$.

2 Background

The Kronecker sum bigraphical lasso (BiGLasso) model was first considered by Kalaitzis et al. [7]. BiGLasso is the multi-axis analog to graphical lasso methods [3, 1], which are used to estimate

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

covariance matrices of data of a multivariate Gaussian distribution. The Kronecker sum of two matrices, $\mathbf{A} \oplus \mathbf{B}$, can be expressed in terms of Kronecker products: $\mathbf{A} \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{B}$. When the matrices $\mathbf{A}, \mathbf{B}$ are adjacency matrices of graphs, the Kronecker sum has the interpretation as the Cartesian product of those graphs[16]. This sum is one choice of function ζ to combine the per-axis precision matrices into the precision matrix of the vectorized dataset, $\mathrm{vec}[\mathbf{D}] \sim \mathcal{N}(\mathbf{0}, (\Psi_{\mathrm{row}} \oplus \Psi_{\mathrm{col}})^{-1})$.

Other choices for ζ have been considered, such as using the Kronecker product[18, 22], or the square of the Kronecker sum[20]. The benefits of a Kronecker sum structure are its interpretability as a graph product, stronger sparsity, and its allowance of inter-task transfer[7].

The original BiGLasso model was very slow to converge to a solution, in large part due to its non-optimal space complexity of $O(n^2 p^2)$. This prohibited its use on large datasets (measuring in a couple hundred samples and/or features). Numerous modifications have been made to the algorithm to improve its speed and achieve an optimal space complexity of $O(n^2 + p^2)$, such as scBiGLasso[10], TeraLasso[4], and EiGLasso[21]. Of these, TeraLasso is notable in that it generalizes to an arbitrary amount of axes, i.e. $\zeta(\mathbf{\Psi}_1, ..., \mathbf{\Psi}_k) = \mathbf{\Psi}_1 \oplus ... \oplus \mathbf{\Psi}_k$. However, all of these algorithms are still iterative. The convergence time can vary greatly depending on the precise values of the data, and in general they struggle to handle datasets with a couple thousand samples and/or features. Our proposed variant, antGLasso [1], does not suffer from these issues, preserves optimal memory complexity, and like TeraLasso can work on datasets with an arbitrary amount of axes.

All algorithms listed rely on a normality assumption. One could manually transform non-normal data to a Gaussian, if they knew the right transform to use. An alternative method is to use the Nonparanormal Skeptic[11], an algorithm which directly estimates the Gram matrices of data that has been transformed to a Gaussian by some unspecified function[10]. Since all previous BiGLasso algorithms mentioned can be framed in terms of accepting these Gram matrices as input, they can circumvent the normality assumption.

3 Methodology and Results

In the following we introduce the main theorems and results of our study. For our implementation, we used Python Version 3.9.12, Numpy Version 1.22.2[5], Scipy Version 1.7.3[19], Sklearn Version 1.0.2[14], Matplotlib Version 3.5.1[6], and Pandas Version 1.4.2[17, 12]. All tests were run on a MacBook Pro (13-inch, M1, 2020) with 8GB of RAM, using Jupyter Notebooks[15, 8].

3.1 Theorems

The general structure of the algorithm is as follows: we first directly estimate the eigenvectors of the solution. We then use the eigenvectors to diagonalize the covariance matrix of the tensor-variate distribution, such that we can directly estimate the eigenvalues from the variances of the data. We follow the notation of Kolda and Bader [9] for tensor operations, and Greenewald, Zhou, and Hero [4] for BiGLasso variables. $\mathbf{K}_b^a$ is shorthand for the matrix with a on the diagonals and b elsewhere.

Theorem (Eigenvectors of Precision Matrices)**.** $\mathbf{V}_\ell$*, the eigenvectors of* $\mathbf{\Psi}_\ell$*, are the eigenvectors of* $\mathbf{S}_\ell \circ \mathbf{K}_{m_\ell}^{2m_\ell - 1}$*, where* $\mathbf{K}_b^a$ *is the matrix with diagonal elements* a *and off-diagonal elements* b*.*

Theorem (Eigenvalues of Precision Matrices)**.** *Let* $\mathbf{a}$ *be the vector of length* $\prod_\ell^K d_\ell$ *whose* $(\sum_\ell i_\ell \prod_{j=1}^{\ell-1} d_j)$*th element is* $\frac{1}{\mathrm{var}[[\mathcal{Y} \times_1 \mathbf{V}_1^T \times_2 ... \times_K \mathbf{V}_K^T]_{i_1 ... i_K}]}$*. Then there is a system of linear equations relating* $\mathbf{a}$ *to the eigenvalues of* $(\mathbf{\Psi}_1, ..., \mathbf{\Psi}_K)$*.*

Unfortunately, the calculation of the eigenvalues requires the whole dataset $\mathcal{Y}$, whereas the use of the Nonparanormal Skeptic requires that the algorithm be parametrizable in terms of the Gram matrices. We can circumvent this by introducing a heuristic, $\mathrm{var}\left[\mathcal{X}_{i_1,...,i_K}\right] \approx d_\ell \Delta_{i_\ell} \mathbf{V}_\ell^T \mathbf{S}_\ell \mathbf{V}_\ell \Delta_{i_\ell}^T$, where Δ_{i_ℓ} is a row vector of zeros except for a one at position i_ℓ. We call this variant of the algorithm 'Heuristic antGLasso'; further details are available in the supplementary material.

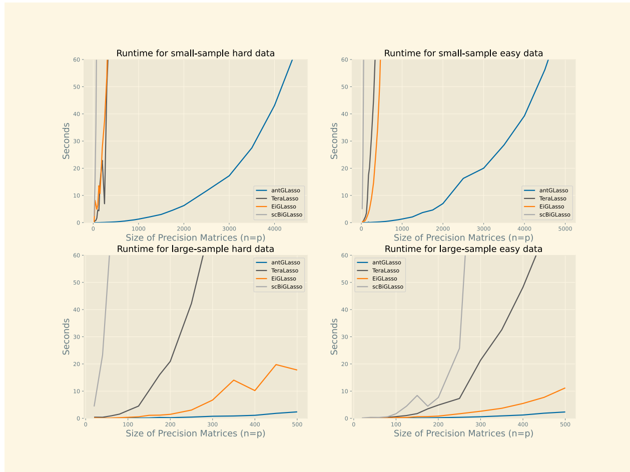

Figure 1: A comparison of runtimes of four BiGLasso algorithms on simulated data. We consider small samples ($s = 1$) and large samples ($s = 100$), as well as data drawn from distributions that tend to cause algorithms to converge slowly (‘hard’) and quickly (‘easy’).

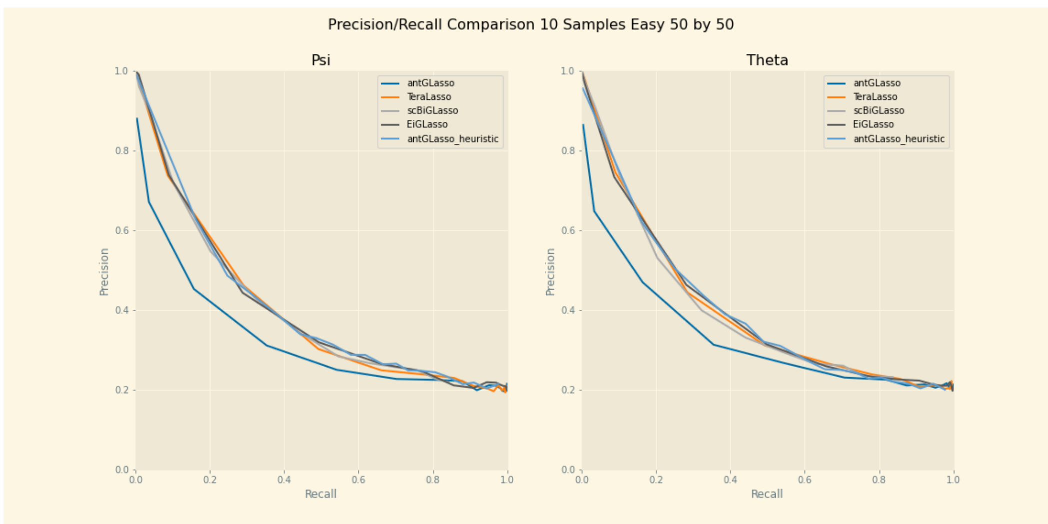

Figure 2: Precision-recall curves for two-dimensional simulated data from a distribution that converged quickly (‘easy’), for 50x50 input matrices.

3.2 Results on Simulated Data

We first tested our algorithms on simulated data. The precision matrices were drawn from an inverse Wishart distribution. The iterative algorithms tended to converge slowly when the degrees of freedom parameter of the distribution was small, and quickly when it was large. Additionally, they converged much slower on small sample data (Figure 1). We can see that antGLasso is by far the fastest algorithm, giving up to two orders of magnitude improvement over the next fastest algorithm (EiGLasso).

Not pictured are the results for tensor data, due to space constraints. For three-axis tensors, antGLasso is still markedly faster than TeraLasso in the small sample case - we did not have enough RAM to test the large sample case. For four-axis and larger, the difference between the algorithms becomes negligible - this is because the major runtime constraint becomes the calculation of the Gram matrices, whose complexity grows exponentially with the amount of axes and thus dominate.

[1] **an**alytic **t**ensor-**g**raphical **lasso**

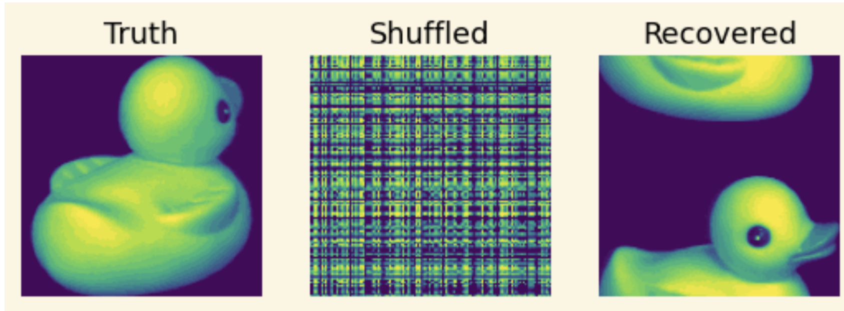

Figure 3: Recovery of a video of a spinning rubber duck. (Left) A frame from the original video, (Middle) a frame from the shuffled video, (Right) frame from the reconstructed video. Not represented in the image is the recovery of the frames of the video, which was perfect.

antGLasso does not perform as well as other algorithms (Figure 2), although this gap drops as the number of samples increase. Interestingly, Heuristic antGLasso performs comparably to the others while preserving the speed of antGLasso. We believe this is because antGLasso directly estimates the variances of the data, which will be inaccurate for small samples. The heuristic does not rely on estimating the variance, so it circumvents this issue. The gap between the two variants drops as the number of samples increases.

3.3 Results on and Applicability to Real World Data

We have shown that on simulated data Heuristic antGLasso performs comparably but much quicker than existing algorithms. This enables the application of BiGLasso methods to datasets in the size of thousands. For real world experiments, let's consider the case of video data. Given the graph of conditional dependencies, we might expect to be able to order the frames using this observation, as a frame should only be conditionally dependant on its nearest neighbors.

For example, consider the video of a rubber duck from the COIL-20 dataset[13]. If we scramble the rows, frames, and columns of the video, we can recover the original video from the outputs of antGLasso, as seen in Figure 3. The reconstruction is perfect, except for the duck being cut in half. This is due to the difficulty in figuring out which row to start on.

The strong results for video reconstruction are encouraging. However, when we modify the video by duplicating frames and adding a bit of noise our reconstruction performs much worse. Already by duplicating each frame 5 times we dip below 10% reconstruction accuracy. This is because duplicating frames interferes with the conditional dependency graph. This suggests that temporal recovery can only be done in highly structured cases with few redundancies. We verified that this behavior is seen in all BiGLasso algorithms considered in the report, including antGLasso.

4 Conclusion

We have created a new BiGLasso algorithm, antGLasso, which is much faster than current competitors, while preserving (in the heuristic case) their accuracy. It can be applied to tensors of arbitrary dimensions, like TeraLasso, and performs well on simulated data as well as highly-structured real-world data. It falters when applied to more complicated data, which is a major limitation. However, this is a systemic problem with BiGLasso algorithms rather than our implementation specifically. In future work, we would like to understand better how to apply this methodology to less structured datasets, such as scRNA-seq data.

References

[1] Onureena Banerjee, Laurent El Ghaoui, and Alexandre d'Aspremont. "Model Selection Through Sparse Maximum Likelihood Estimation for Multivariate Gaussian or Binary Data".

In: *Journal of Machine Learning Research* 9.15 (2008), pp. 485–516. URL: `http://jmlr.org/papers/v9/banerjee08a.html`.

[2] Andy Dahl et al. *Network inference in matrix-variate Gaussian models with non-independent noise*. 2013. DOI: `10.48550/ARXIV.1312.1622`. URL: `https://arxiv.org/abs/1312.1622`.

[3] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *Sparse inverse covariance estimation with the lasso*. 2007. DOI: `10.48550/ARXIV.0708.3517`. URL: `https://arxiv.org/abs/0708.3517`.

[4] Kristjan Greenewald, Shuheng Zhou, and Alfred Hero. *Tensor Graphical Lasso (TeraLasso)*. 2017. DOI: `10.48550/ARXIV.1705.03983`. URL: `https://arxiv.org/abs/1705.03983`.

[5] Charles R. Harris et al. "Array programming with NumPy". In: *Nature* 585.7825 (Sept. 2020), pp. 357–362. DOI: `10.1038/s41586-020-2649-2`. URL: `https://doi.org/10.1038/s41586-020-2649-2`.

[6] J. D. Hunter. "Matplotlib: A 2D graphics environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. DOI: `10.1109/MCSE.2007.55`.

[7] Alfredo Kalaitzis et al. "The Bigraphical Lasso". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research 3. Atlanta, Georgia, USA: PMLR, 17–19 Jun 2013, pp. 1229–1237. URL: `https://proceedings.mlr.press/v28/kalaitzis13.html`.

[8] Thomas Kluyver et al. "Jupyter Notebooks - a publishing format for reproducible computational workflows". In: *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. Ed. by Fernando Loizides and Birgit Scmidt. Netherlands: IOS Press, 2016, pp. 87–90. URL: `https://eprints.soton.ac.uk/403913/`.

[9] Tamara G. Kolda and Brett W. Bader. "Tensor Decompositions and Applications". In: *SIAM Review* 51.3 (Sept. 2009), pp. 455–500. DOI: `10.1137/07070111X`.

[10] Sijia Li et al. "Scalable Bigraphical Lasso: Two-way Sparse Network Inference for Count Data". In: (Mar. 2022).

[11] Han Liu et al. *The Nonparanormal SKEPTIC*. 2012. DOI: `10.48550/ARXIV.1206.6488`. URL: `https://arxiv.org/abs/1206.6488`.

[12] Wes McKinney. "Data Structures for Statistical Computing in Python". In: *Proceedings of the 9th Python in Science Conference*. Ed. by Stéfan van der Walt and Jarrod Millman. 2010, pp. 56–61. DOI: `10.25080/Majora-92bf1922-00a`.

[13] Sameer A. Nene, Shree K. Nayar, and Hiroshi Murase. *Columbia Object Image Library (COIL-20*. Tech. rep. 1996.

[14] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[15] Fernando Pérez and Brian E. Granger. "IPython: a System for Interactive Scientific Computing". In: *Computing in Science and Engineering* 9.3 (May 2007), pp. 21–29. ISSN: 1521-9615. DOI: `10.1109/MCSE.2007.53`. URL: `https://ipython.org`.

[16] Gert Sabidussi. "Graph Multiplication." In: *Mathematische Zeitschrift* 72 (1959/60), pp. 446–457. URL: `http://eudml.org/doc/183624`.

[17] The pandas development team. *pandas-dev/pandas: Pandas*. Version latest. Feb. 2020. DOI: `10.5281/zenodo.3509134`. URL: `https://doi.org/10.5281/zenodo.3509134`.

[18] Theodoros Tsiligkaridis and Alfred O. Hero. "Covariance Estimation in High Dimensions Via Kronecker Product Expansions". In: *IEEE Transactions on Signal Processing* 61.21 (Nov. 2013), pp. 5347–5360. DOI: `10.1109/tsp.2013.2279355`. URL: `https://doi.org/10.1109%2Ftsp.2013.2279355`.

[19] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: `10.1038/s41592-019-0686-2`.

[20] Yu Wang, Byoungwook Jang, and Alfred Hero. "The Sylvester Graphical Lasso (SyGlasso)". In: *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. Ed. by Silvia Chiappa and Roberto Calandra. Vol. 108. Proceedings of Machine Learning Research. PMLR, 26–28 Aug 2020, pp. 1943–1953. URL: `https://proceedings.mlr.press/v108/wang20d.html`.

[21] Jun Ho Yoon and Seyoung Kim. "EiGLasso: Scalable Estimation of Cartesian Product of Sparse Inverse Covariance Matrices". In: *Proceedings of the 36th Conference on Uncertainty in Artificial Intelligence (UAI)*. Ed. by Jonas Peters and David Sontag. Vol. 124. Proceedings of Machine Learning Research. PMLR, Mar. 2020, pp. 1248–1257. URL: `https://proceedings.mlr.press/v124/ho-yoon20a.html`.

[22] Shuheng Zhou. "Gemini: Graph estimation with matrix variate normal instances". In: *The Annals of Statistics* 42.2 (Apr. 2014). DOI: `10.1214/13-aos1187`. URL: `https://doi.org/10.1214%2F13-aos1187`.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes] We claim that our algorithm is faster than competitors, and give evidence to this in Figure 1.
   (b) Did you describe the limitations of your work? [Yes] As mentioned in the Results section, non-heuristic variant performs poorly on small sample data and BiGLasso algorithms in general perform poorly when data gets more complicated, such as duplicated video frames
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] In the supplementary material.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes] Code that runs experiments discussed in this paper and the supplementary material is available as part of the supplementary material.
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] In the supplementary material we talk about hyperparameters, and in the metrics section we talk about the values we chose.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes] antGLasso's performance curves in the Regularization section of the supplementary material highlight the variance.
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] This is Figure 1.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]
(b) Did you mention the license of the assets? [N/A]
(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]